A BYTE BOOK

>>$9.95

MUNCHERS
The Ultimate TIMEX/SINCLAIR Games Book

Yin Chiu & Henry Mullish



For the **T/S 1000, 1500,** and **2068** Computers

MUNCHERS

The Ultimate Timex/Sinclair Games Book

Yin Chiu and Henry Mullish

McGraw-Hill Book Company

New York St. Louis San Francisco Bogotá Guatemala
Hamburg Lisbon Madrid Mexico Montreal New Delhi
Panama Paris San Juan São Paulo Singapore Tokyo Toronto

The authors of the programs provided with this book have carefully reviewed them to ensure their performance in accordance with the specifications described in the book. Neither the authors nor McGraw-Hill, however, makes any warranties whatever concerning the programs. They assume no responsibility or liability of any kind for errors in the programs or for the consequences of any such errors.

As used in this book, the terms "Timex/Sinclair 1000, 1500," and "2068" and "T/S 1000, 1500," and "2068" refer to computers manufactured and sold by the Timex Computer Corporation. Timex/Sinclair 1000, Timex/Sinclair 1500, and Timex/Sinclair 2068 are registered trademarks of the Timex Corporation.

A BYTE Book.

2 3 4 5 6 7 8 9 10 S E M S E M 8 7 6 5 4

ISBN 0-07-010839-0

Library of Congress Cataloging in Publication Data

Chiu, Yin
Munchers : the ultimate Timex/Sinclair games book.

1. Computer games. 2. Timex/Sinclair 1000 (Computer)—Programming. 3. Timex/Sinclair 1500 (Computer)—Programming. 4. Timex/Sinclair 2068 (Computer)—Programming. 5. Basic (Computer program language) I. Mullish, Henry. II. Title.
GV1469.2.C475 1984 794′.8 84-10028
ISBN 0-07-010839-0

The editors for this book were Jeff McCartney and Karl Weber; the editing supervisor was Margery Luhrs.

PREFACE

It was in the summer of 1982 that the Timex/Sinclair Corporation broke through what was then considered an impregnable barrier by selling a computer on the retail market for less than $100. Even though its memory was restricted to 2K (about 2000 characters) the computer sold by the hundreds of thousands, and this sales volume enabled the company to offer further substantial price reductions. On the heels of this unprecedented success, the company released in November 1983 their sleek, superior model 1500, which has an attractive keyboard and eight times the memory capacity of the T/S 1000. The model 1500 comes with a built-in 16K memory and sells, like its predecessor, for under $100 (in some cases, well under $100). Then in January of 1984, the Timex/Sinclair 2068 was introduced with a 72K memory, and color and sound capabilities. Remarkably, it sells for less than $200.

No longer can it be said that computers are only for the wealthy. Now any kid can wash a few cars, mow a few lawns, or babysit for a few evenings to save sufficient funds to go out and buy a personal computer requiring no other additional equipment than a black-and-white television set. In his or her hands will be the same computational power that, until a mere generation ago, was reserved for giant corporations, and prestigious educational and scientific institutions. At a time when there is much concern about increasing the nation's computer literacy, the inexpensive, lightweight, friendly Timex/Sinclair computers answer the call by providing an ideal introduction to the world of computers.

The programs that we have devised take advantage of the popularity of video arcade games. Of the 32 original games that we have written, many are of the space-invader type. These will, no doubt, appeal to those who are tired of throwing

their hard-earned quarters into arcade machines. In addition to these are other games that will provide more than just entertainment. They are designed so they will improve certain physical and intellectual skills. They will be both fun and a challenge to your wits and memory. All of the games are programmed in the most popular of the home computer languages—BASIC. Not only will you be able to amuse yourself and others with these games once they are entered into the machine, you will be able to follow the logic on which they are based with the detailed explanation that accompanies each of the programs. We also suggest various ways in which the games may be customized to suit your own needs.

Great care has been taken to print the programs and instructions as clearly as possible. In particular, all references to Timex/Sinclair keywords are printed in bold type to alert the person typing the programs into the computer that the word in question is accessed by a single stroke of the selected key. At the end of each chapter, we have provided suggestions for possible additions and changes for each of the programs. In this way, the games may be tailored to individual needs and keep their freshness throughout the life of the computer. For the novice, explicit keyboarding instructions are given on how to save, edit, and run programs, how to save programs on a tape recorder, and how to load programs into the computer from tape.

The task of writing games for the computer is not an easy one. What human beings take for granted has to be spelled out in the minutest detail for the computer. Often quite intricate approaches must be adopted to achieve the desired goal. Imagine for a moment what would be involved in relating to a computer the enormous number of details that a child readily deals with when playing a simple game of dominoes. The explicit instructions that a computer must receive to perform this simple process reveals the limits of programming. Infinite patience and thoroughness are necessary. After all, a computer is an inanimate object. It is a cleverly designed piece of equipment, but it is still composed only of plastic, wires, and a little metal.

It is our wish that after you have had an opportunity to

become thoroughly familiar with the programs you will absorb many of the techniques and methods to help you write your own games. This is an intellectual challenge that should not be minimized. Whatever expertise is gained in this endeavor can be put to use in regular programming, whether the discipline involved be mathematics, physics, chemistry, finance, education, or engineering.

This book has been designed for all owners of Timex/Sinclair computers. The book is divided into three parts: Part I contains games for the T/S 1000, Part II has games for the T/S 1500, and Part III has games for the T/S 2068. If you have a T/S 1000, you can program and play all of the games in the first section of the book. To play more sophisticated games, however, you will need some more memory. You can purchase a 16K RAM pak for your T/S 1000, or the more advanced T/S 1500 model (with 16K) to play all of the games in Parts I and II. The games in Part III were written to use the color and sound capabilities of the T/S 2068; of course, this latest Timex/Sinclair model can easily handle all of the programs in both Parts I and II.

If this is your first introduction to game playing on a computer, welcome, and lots of luck!

Yin Chiu
Henry Mullish

CONTENTS

Chapter 0

Getting Started

- How to Use This Book
- Setting Up the Computer
- Keyboarding Hints

Even if you have already had some experience in using the Timex/Sinclair computers, you should take a few minutes to read this chapter. The T/S computers are remarkably powerful for their price, but to use them to their best advantage you must be familiar with their particular dialect of BASIC, perhaps the most widely used computer language in the world today, and you must be familiar with the T/S keyboards, which are organized considerably differently from the keyboards found on most other computers.

A brief reading of the points covered in this chapter, along with the instructions that appear in the *Timex/Sinclair User Manual* that came along with your computer, should enable you to set up the computer and enter the first program within one hour.

How to Use This Book

At the heart of this book are the 32 games programs, with the first program listed in Chapter 1 and the thirty-second listed in Chapter 32. We strongly recommend that you enter the first program before jumping around in the sequence to select others. We make this recommendation because the first program is very clearly annotated, and contains explanations of many of the typographic conventions that we use in this book. In subsequent programs we assume that you have mastered the

basics of the keyboard, so that the annotations are much fewer and further between.

Each chapter in this book begins with an explanation of the game that you will get when you run the program. You can get an idea of just what the game consists of by reading this section, called simply ''The Game.'' Following the description of the game is the actual listing of the program, and in Chapter 1 the listing is followed by ''Keyboarding Notes,'' which are meant to provide the absolute beginner with instructions on how to manipulate the T/S keyboard.

Chapter 1 also contains instructions on how to edit a program and run a program. Neither of these activities is difficult, but if you have done neither before, you should read these sections carefully.

All the chapters contain two additional sections: ''Points of Interest'' and ''Possible Adaptations.'' The material under ''Points of Interest'' is intended as an explanation of the logic of the programs. If you are an absolute beginner, these ''points'' will be somewhat less helpful, but by studying them in conjunction with the program you will begin to get an inkling of how BASIC works.

The ''Possible Adaptations'' at the end of each chapter either give specific instructions on how to alter a program (for the absolute beginner) or *suggest* ways in which a program can be customized (for more advanced readers).

Understanding a computer program is a serious intellectual challenge, but it can also be a lot of fun. We hope that you will use the instructions in this book to tinker with, revise, expand, and customize these programs. Before very long you'll realize that you are truly ''getting into'' these programs, and in a very real sense, you'll be making them your own. Remember that no matter what keys you press, there is absolutely no way to damage the computer. In that sense it is foolproof.

Setting Up the Computer

Instructions for setting up your computer appear in the first part of the *Timex/Sinclair User Manual* that came with your model. These instructions, together with the many excellent

diagrams, are fairly straightforward, and you should not have very much difficulty in following them. We have found, however, that a few additional adjustments to the setup can make your system even easier to use.

1. *The cable connecting the T/S 1000 or T/S 1500 to the TV:* The manufacturer has supplied a 4-foot cable that connects your computer and the TV. We're found that in terms of convenience and comfort this cable is much too short. We recommend that you replace the 4-foot cable with a 6- or 8-foot cable. (Most electronics or television and radio stores will have such cables in stock, and one shouldn't cost more than a few dollars.)

   For the most part the T/S 1000 and 1500 produce a TV image with black characters on a white background, so the screen is quite bright. With the 4-foot cable supplied with the computer by the manufacturer, we found ourselves sitting almost on top of the screen, and after a few hours of work we were suffering from eye strain and beginning to feel a little bit spaced out. With a longer cable, we were able to move the TV to the far end of a table. In such a setup, we still found the image on the screen to be perfectly readable, and it was certainly more restful for our eyes.

   An 8-foot cable is already provided with the T/S 2068 model.

2. According to the *Timex/Sinclair User Manual,* the TV should be set to either channel 2 or channel 3, whichever one is not being used for broadcasting in the local area. This is not always the best advice. We have found that it is sometimes better to set the TV to the same channel that broadcasts in the local area. If you do not set the channel at all, it defaults to channel 2. So once all the electrical connections have been made, see if channel 2 gives you good reception. If not, select channel 3; use the channel selection switch on T/S 1000 or 2068 models to indicate your choice. You

can change channels on the T/S 1500 by pressing the key on the keyboard marked 3 before commencing. In New York City, where channel 2 is used by the CBS-TV affiliate, channel 2 seemed to work better than channel 3, which is not used for broadcasting purposes.

3. *Setting up your tape recorder:* First let it be said that having a tape cassette recorder makes a world of difference so far as convenience of entering programs is concerned. Once a program has been typed into the computer it may be saved on tape; from that point on, all you have to do is load in the tape to rerun that program. This avoids the tedium of having to retype in the program. Even if the program is short, the probability of making errors is high. Naturally, the probability is greater with lengthy programs.

   We recommend that the "volume" control of the tape recorder be at or near its loudest setting. Similarly, better results might be obtained when loading and saving programs if the "tone" control is at its highest setting—toward "treble" rather than "bass." Furthermore, if your tape recorder has both "monaural" and "stereo" modes, use the "monaural" setting. If a stereo setting is all that is available, you should turn the "balance" control all the way to the left or all the way to the right and leave it there for both loading and saving to the tape recorder.

Three further points: First, we've found that it pays to use a fairly high quality recording tape. Second, it is a good idea to make a backup copy of each program, preferably on a completely different cassette. And third, you should record programs on one side of the tape only.

Keyboarding Hints

If you haven't already had some experience with the T/S keyboards, you should read this section carefully. As you read it,

you may want to refer to the illustration of the keyboards in Appendix D at the end of the book. *(Courtesy of Timex Computer Corporation.)*

1. When you are typing in the number 1, be sure to hit the 1 key and not the L key, as you might be tempted to do if you have not had much typing experience. The 1 key is located at the top row of the keyboard at the extreme left.
2. Don't confuse the number 0 with the letter O. In program listings in this book and on the screen of your TV, you'll notice that zero is printed with a slash through it, like this: Ø. This is to make it clear to you which of the two symbols is intended.
3. The letter I and the number 1 may also look alike to you as you read the program listings, but you must be sure to distinguish between the two. The letter I is shown in programs this way: I. The number 1 is listed in the programs this way: 1.
4. In some of the programs we make use of an underscore (_) to indicate that you should leave a space (by hitting the space key). Do not attempt to key in this symbol; you will not find it on the T/S 1000 or 1500 keyboard. (The minus sign is there but it prints in the middle of the line rather than at the bottom.) You *will,* however, find the underscore key on the T/S 2068 keyboard, but make sure you don't key it in; it is not used in any of these programs. We make use of this symbol only within quotation marks in the program listing because it is only within quotation marks that inserting the proper number of spaces may be critical.
5. In this book, the only quotation marks we use are the ones located on the P key. (For *none* of our 32 programs should you use the other quotation marks, which are located on the Q key.)
6. All words printed in **BOLDFACE** in the program listings are called keywords. Each of these words is

accessed by a single keystroke. If you try to type in one of these words letter-by-letter, the computer will either not accept it or will not run the program correctly. So whenever you see a boldface word in a listing, be sure to access it with a single keystroke. Incidentally, even if you have the incredibly efficient Timex/Sinclair model 2040 printer to list your programs for you, the keywords are printed in ordinary capital letters. (A list of the T/S keywords and how to access them is contained in Appendix B at the back of the book.)

7. All of the following mathematical symbols are also accessed with a single keystroke:

   < > which means "not equal to" and is found on the T key on the T/S 1000 and 1500, or the W key on the T/S 2068.

   >= which means "greater than or equal to" and is found on the Y key on the T/S 1000 and 1500, or the E key on the T/S 2068.

   <= which means "less than or equal to" and is found on the R key on the T/S 1000 and 1500, or the Q key on the T/S 2068.

8. Keep in mind that any given key on the keyboard may have as many as five meanings, depending upon the *mode* you are in. The mode is indicated by a cursor, as follows:

   K indicates the keyword mode. When you are in this mode, you will access the word printed in white above the key.

   L indicates the letter mode. When you are in this mode, you will access the "primary" character on the key.

   C indicates the uppercase mode for the T/S 2068. This mode is similar to the **L** mode of the T/S 1000 and 1500 models, but is more like the shift-lock key of a typewriter. The T/S

2068 has the added feature of being able to print lowercase letters along with the usual uppercase letters.

F indicates the function mode on the T/S 1000 and T/S 1500. When you are in this mode, you will access the word printed in white below the key.

E indicates the function mode of the T/S 2068. When you are in this mode, you will access the small word printed in white on the key.

G indicates the graphics mode. On the T/S 1000 and T/S 1500, if you hit a key in this mode, the "primary" character will appear in inverse video. (This means that the character will appear on the screen as a white number or letter superimposed on a black background.) On all three T/S computers, in the graphics mode, some of the keys will give you the small graphics symbol shown at the lower right corner of the key. For a list of these graphics symbols and how to access them, see Appendix C at the end of this book.

PART I

Games for the Timex/Sinclair 1000

The five games in this part were written primarily for use on the first of the Timex/Sinclair computers, namely the T/S 1000. The programs do not exceed 2K in length and, therefore, do not require the T/S 1000 RAM pak. Naturally, they can run on the T/S 1500 and T/S 2068 as well, but the T/S 2068 display will be in black and white.

Chapter 1

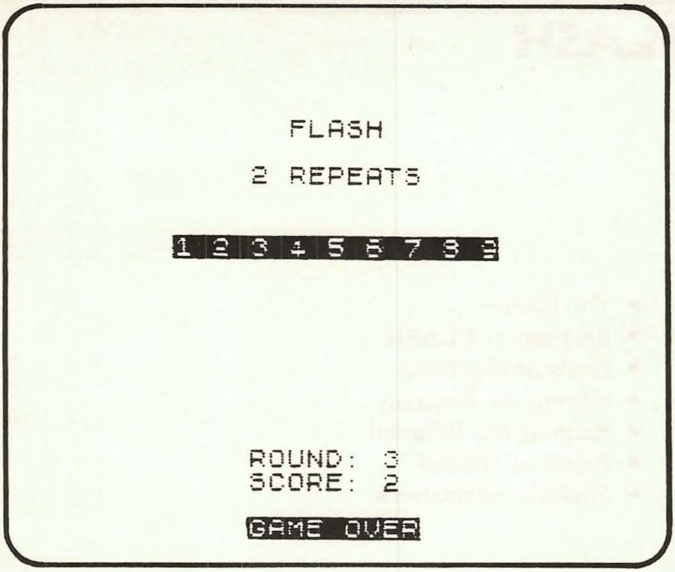
FLASH

- The Game
- Program 1: FLASH
- Keyboarding Notes
- Editing the Program
- Running the Program
- Points of Interest
- Possible Adaptations

The first game we will present is called "Flash." We've chosen to present it first not because it is the most interesting game in this book, but because it is one of the easier programs to understand. Whether you are a complete novice to computer programming or an old hand at it, the chances are good that you will derive a good deal of useful information from our step-by-step discussion of this program.

The Game

The object of the game is to test your alertness and the quickness of your eye. When the game is "run" on the computer, a row of digits 1 to 9 (in inverse video) is permanently displayed on the upper half of the screen. The computer momentarily flashes random digits under their corresponding inverse images. Only one of those digits is flashed one time more than any of the others. Exactly how many times it is flashed is indicated on the "number of repeats" line which appears just above the inverse row of digits. The trick is to determine which one flashed exactly that number of times.



The number of flashes increases as the game proceeds, so it becomes difficult very early in the game. Whatever you do, don't divert your attention from the screen. You might want to adjust the screen to enhance the clarity of the flashes because it goes by very quickly. Four correct responses in a row permits you to proceed to the next level of difficulty. An incorrect response, on the other hand, terminates the game at that point and you are informed that the game is over (displayed in inverse video—if that makes you feel any better).

What follows is a listing of the program. Take a few minutes now to look it over and then follow the explicit step-by-step instructions that appear just after the program listing. Within a few minutes you'll find that you are on your way to becoming an experienced keyboarder.

Program 1: FLASH

```
100  DIM X(9)
110  LET L$="1 2 3 4 5 6 7 8 9"
```

```
115  LET S=Ø
12Ø  LET N=1
13Ø  GOSUB 1ØØØ
135  LET G=1
14Ø  GOSUB 2ØØØ
15Ø  LET Z=N*9+1
16Ø  LET M=INT (RND*9)+1
17Ø  LET X(M)=-1
18Ø  FOR K=1 TO Z
19Ø  LET D=INT (RND*9)+1
2ØØ  IF X(D)=N THEN GOTO 19Ø
21Ø  LET P=(D-1)*2+7
22Ø  PRINT AT 1Ø,P;D;AT 1Ø, P;"_"
23Ø  LET X(D)=X(D)+1
24Ø  NEXT K
3ØØ  PRINT AT 15,1Ø;"WHICH_ONE_?"
31Ø  INPUT W
315  PRINT AT 15,1Ø;"___________"
32Ø  IF W<>M THEN GOTO 3ØØØ
33Ø  LET S=S+1
335  LET G=G+1
34Ø  IF N<5 AND G=5 THEN LET N=N+1
345  PRINT AT 17,18;S+1
35Ø  PRINT AT 18,18;S
355  IF G=5 THEN GOTO 135
36Ø  GOTO 14Ø
1ØØØ CLS
1Ø1Ø PRINT AT 3,13;"FLASH"
1Ø2Ø PRINT AT 8,7;L$
```

```
1Ø3Ø  PRINT AT 17,11;"ROUND:_1"
1Ø4Ø  PRINT AT 18,11;"SCORE:_Ø"
1Ø5Ø  RETURN
2ØØØ  FOR I=1 TO 9
2Ø1Ø  LET X(I) =Ø
2Ø2Ø  NEXT I
2Ø3Ø  PRINT AT 5,11;N+1;"_REPEATS"
2Ø4Ø  RETURN
3ØØØ  FOR K=1 TO 1Ø
3Ø1Ø  PRINT AT 2Ø,11;"GAME_OVER";AT 2Ø, 11;
      "GAME OVER"
3Ø2Ø  NEXT K
```

Keyboarding Notes

Refer to the above program "FLASH," and mark it with your own notes as you follow these steps.

1. Switch on the computer and be sure that the K cursor is displayed in the lower left corner of the screen.
2. As you are probably aware, all BASIC statements must begin with a statement number or line number. Since we have arbitrarily assigned the line number 100 to the first line of the program, type in 1ØØ. You will note that the K cursor remains on the screen after 100 has been typed in.
3. After the statement or line number 100, you will want to type in the keyword **DIM**, which is printed in **boldface** in this book's listing of the program. The boldface listing indicates, as we mentioned in Chapter 0, that the word in question should be entered with a single keystroke, and not letter-by-letter. Since you are still in the K mode, you can access the keyword **DIM** simply by hitting the D key, after which the L cursor will automatically appear.

4. Now type in the letter X, located to the left of the third (bottom) row of letter keys. This is followed by the opening parenthesis (on the I key), the number 9, and, finally, the closing parenthesis (on the 0 key). In order to get at the parentheses you must use the SHIFT key, the one on the extreme left side of the lower row of keys. With one finger hold down the SHIFT key and, at the same time, press the I key (for the left parenthesis) with a finger of the other hand. This operation must be done simultaneously. It will not work if you press these keys in succession—they must be pressed at one time. We will indicate this dual-feature in the future by writing, for example, shifted-I. The same procedure applies to the right parenthesis, except that this time you will access the character by shifted-O.

   Once the whole line is typed in correctly, you must press the ENTER key in order to send it to the computer. If (heaven forbid!) you made a mistake in typing in the line, the **S** cursor, indicating a syntax error, may appear at the end of the line. In that case the computer's built-in checking system will not accept the line, and the best thing to do is to hit the DELETE key (shifted-Ø) as many times as necessary to remove the line. You can then start again.

   On the assumption that the line was correctly typed in, the line is transferred to the top of the screen with the **>** cursor appearing after line number 100. (This symbol in reverse video is used on the computer as an edit cursor, about which we shall have something to say a little later on.) Note that you are automatically back in the **K** mode.

5. The next line is a **LET** statement—officially called an assignment statement. It has been given the line number 110, so this has to be typed in before the statement. Like the **DIM** statement in line 100, the word **LET** is a keyword. It is accessed by the single keystroke L, since the machine is still in **K** mode. Once a keyword is typed, the machine goes into the **L** mode

and so the variable, L$, can be typed in. The dollar sign, residing on the U key and indicated in red, is accessed by using the SHIFT key. The next symbol needed is the equal sign, =, which is found on the L key. To access it, type shifted-L. The double quote sign, ‘‘, is also printed in red so it too is accessed with the SHIFT key. (By the time you have finished typing in this program you should become quite expert at using the SHIFT key!) Make sure you use the double quote sign—the one on the P key, not the paired quote sign on the Q key.

6. It is now time to type in the row of digits 1 through 9 in inverse video. In order to achieve this effect, all that is necessary is to switch the computer into graphics mode. To do this simply press shifted-9. This displays the G cursor. Now type in the digits 1 through 9 with one space separating each one of them. To exit from graphics mode once the digits have been typed in inverse video, you may either hit the shifted-9 key again or the ENTER key. This permits the closing quote to be typed in when the L cursor reappears.

7. Once you have sent line 110 to the top of the screen by depressing the ENTER key, you are ready to tackle the rest of the program. Lines 115 and 120 are quite straightforward but line 130 may look somewhat strange to you since there is no equivalent phrase in English. The **GOSUB** command is a reference to a structure called a subroutine which you will encounter in your study of BASIC programming. It is an extremely useful command, so if you don’t already understand how it works perhaps this would be a good opportunity to learn. Since it is a keyword, only one key has to be pressed to obtain it. Which one? You’re so right, the H key.

8. Some of the statements take more than one line to type in. This is perfectly legal. Line 315 is an example of one that extends beyond one line. Following the open quote sign there are 11 blanks, which we have

indicated by 11 consecutive underbars. Remember, don't look for the underbar symbol on the screen; it isn't there. We just want to give you a sure way of counting exactly how many spaces there are. The closing quote has no choice but to appear at the beginning of the next line which does not take a line number. Occasionally, the line will be listed slightly different from the way it was typed in. The computer tries to get it all on a single line.

9. You shouldn't have much difficulty with the rest of the program, so go ahead and see how well you can do. Don't forget to hit the ENTER key at the end of each statement.

Once you have finished entering the program, look it over for errors because we are now going to learn how to edit. Are you ready?

Editing the Program

Suppose that while typing in a line you make a mistake. Rest assured that this happens to *every* programmer, no matter how smart or careful he or she is. So *when* it happens to you (notice we didn't say *if* it happens to you), don't panic—it's par for the course.

If you've completely forgotten to enter a line, you can simply type that line at the bottom of the screen and hit the ENTER key. It will automatically insert itself in its proper place in the program. You can always retype a line that has been incorrectly entered; when you ENTER the retyped line, it will automatically "bump" the incorrect line from the program. In actuality, it replaces the previous line.

If when typing in a character you realize immediately that you hit a wrong key, the DELETE key (shifted-Ø) will erase the last character or keyword and reposition the cursor to the preceding location. This enables you to type in the correct character. The original one disappears for all time, and nobody need ever know about it.

Suppose, however, that a line has already been entered into the listing and you now decide that you want to change

it in some way or other. If it is the current line that you want to change (the ▶ cursor signifies which line is current), hit the EDIT key (shifted-1). This brings down a copy of the line to the bottom of the screen so that it may be changed. If the line in the listing does not have the cursor next to it, simply hit LIST and then the line number and ENTER. This will bring the line you want to the top of the screen with the ▶ cursor next to it. Now it is the "current' line and you can bring it down by hitting the EDIT key (shifted-1).

Once a line is at the bottom of the screen and ready to be edited, you can type shifted-8. This has the effect of moving the cursor to the right. (Shifted-5 moves the cursor to the left.) Stop at the appropriate place and make the changes you want. Then, when the ENTER key is pressed, the edited line returns to its proper place in the program.

Running the Program

Now most of the work has been done and it is time to execute the program. Make one last check of the program on the screen, comparing it carefully with the listing in this book. If they appear to match, hit the R key, which in the K mode causes the **RUN** command to come up on the screen. Then, by hitting the ENTER key, you will start the program running.

Enjoy! And if you like "FLASH" enough to want to save it, turn to Appendix A in this book, "Saving Your Programs on a Tape Recorder."

Points of Interest

Line 110: This line sets up the row of digits 1–9 in inverse video and stores it in the variable L$.

Line 150: This line determines how many times to flash the digits.

Lines 160–170: These lines set up which digit is to be repeated one time more than the previous number of repetitions.

Lines 180–240: This series prints out the flashing display and generates a random integer between 1 and 9.

Lines 300–340: These lines prompt the player for the response, check the answer, tally the score, and check to see if the level of difficulty should be raised.

Lines 245–360: This series updates the score and the number of rounds on the screen.

Lines 1000–1050: These lines print the header.

Lines 2000–2040: These lines reset the X array to Ø as well as display how many repetitions to look for.

Lines 3000–3020: These lines flash the "GAME OVER" display in inverse video.

Possible Adaptations

1. If the flashes go by too quickly, a **PAUSE** instruction can be included to slow up the action.
2. The game could be amended so that a single error does not disqualify the player immediately.

Chapter 2

MASTERMIND

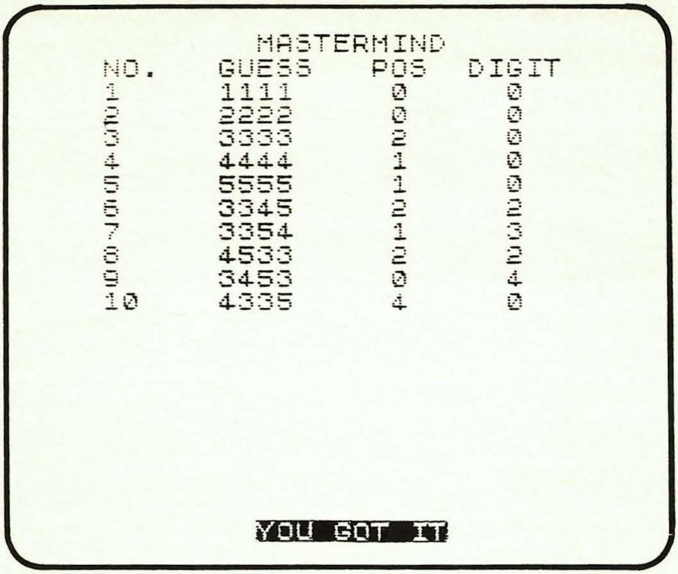
- The Game
- Program 2: MASTERMIND
- Points of Interest
- Possible Adaptations

The Game

This is a version of the famous and equally tantalizing game of Mastermind that vies with the game of chess in its intellectual demands. The computer selects four random digits from the sequence 1 through 6. (The digits can repeat.) The player is asked to determine in as few trials as possible what digits have been selected and in what order. The game has to be completed within 25 attempts in order to win. For each guess made by the player, points are awarded based on the number of digits in the correct position and the number of correct digits. From an analysis of this cryptic information, a shrewd player can get some idea of the hidden digits and what order they're in. As we said, it isn't easy.

If, by accident, more than four digits are entered by the player, the excess digits are ignored and no error is registered. However, if the number of digits entered is less than four, an OUT OF BOUNDS error is triggered and the game is aborted.

If the screen becomes full, the screen is automatically scrolled by the program, allowing for further entries to appear at the bottom of the screen. (The most recent entries on the screen are preserved so that the player can extract the maximum advantage from them.) Care must be taken in inputting data because the player's entries are not validated.



Program 2: MASTERMIND

```
1ØØ  LET X$="123456"
1Ø5  DIM W$(4)
11Ø  FOR I=1 TO 4
12Ø  LET W$(I)=X$(INT (RND*6)+1)
13Ø  NEXT I
14Ø  CLS
15Ø  GOSUB 1ØØØ
16Ø  LET C=Ø
17Ø  FOR J=2 TO 2Ø
18Ø  GOSUB 2ØØØ
19Ø  GOSUB 3ØØØ
2ØØ  NEXT J
21Ø  LET J=2Ø
```

```
220  SCROLL
230  GOSUB 1ØØØ
240  GOSUB 2ØØØ
250  GOSUB 3ØØØ
26Ø  IF C=25 THEN GOTO 4ØØØ
27Ø  GOTO 22Ø
1ØØØ PRINT AT Ø,3;"_ _ _ _ _ _ _ _ _ _ _ _ _ _ _
     _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
1Ø1Ø PRINT AT Ø,11;"MASTERMIND";AT
     1,3;"NO.";TAB (9);"GUESS";TAB (17);"POS";TAB
     (22);"DIGIT"
1Ø2Ø RETURN
2ØØØ PRINT AT 21,1Ø;"YOUR_GUESS_?"
2Ø1Ø INPUT G$
2Ø2Ø PRINT AT 21,1Ø;"_ _ _ _ _ _ _ _ _ _ _ _ _ _"
2Ø3Ø RETURN
3ØØØ LET T$=W$
3Ø2Ø LET H$=G$
3Ø3Ø LET M=Ø
3Ø4Ø FOR Z=1 TO 4
3Ø5Ø IF T$(Z)< >H$(Z) THEN GOTO 3Ø9Ø
3Ø6Ø LET M=M+1
3Ø7Ø LET H$(Z)="_"
3Ø8Ø LET T$(Z)="X"
3Ø9Ø NEXT Z
31ØØ LET N=Ø
311Ø FOR Z=1 TO 4
312Ø FOR Y=1 TO 4
313Ø IF H$(Z)< >T$(Y) THEN GOTO 316Ø
```

```
3140  LET N=N+1
3150  LET T$(Y)="X"
3155  GOTO 3170
3160  NEXT Y
3170  NEXT Z
3180  LET C=C+1
3190  PRINT AT J,3;C;TAB (9);G$;TAB (18);M;TAB
      (24);N
3200  IF G$< >W$ THEN RETURN
3210  FOR Z=1 TO 10
3220  PRINT AT 21,11;"YOU_GOT_IT"
3230  PRINT AT 21,11;"YOU GOT IT"
3240  NEXT Z
3250  PRINT Q
4000  FOR Z=1 TO 10
4010  PRINT AT 21,5;"25_GUESS_LIMIT_REACHED"
4020  PRINT AT 21,5;"25 GUESS LIMIT REACHED"
4030  NEXT Z
```

Points of Interest

Lines 100–130: These lines build the code randomly from the six characters in line 100.

Lines 170–200: This series displays the first 18 guesses before the screen becomes full.

Lines 210–270: These lines control the scrolling of the screen while, at the same time, check for the guess limit of 25.

Lines 1000–1020: These lines refresh the header after the screen has scrolled.

Lines 2000–2030: These lines prompt the player for the guess.

Lines 3000–3200: These lines process the player's guess.

Lines 3210–3250: This set displays the flashing message "YOU GOT IT" when the hidden code is correctly guessed.

Lines 4000–4300: These lines flash the message "25 GUESS LIMIT REACHED".

Possible Adaptations

1. Any symbols at all may be used rather than the digits 1 through 6 in the string X$.
2. The number of guesses allowed may be changed from 25 to any number of your choosing.
3. You might decide not to allow for the repetition of the digits in the code.

Chapter 3

ALIEN ZAPPER

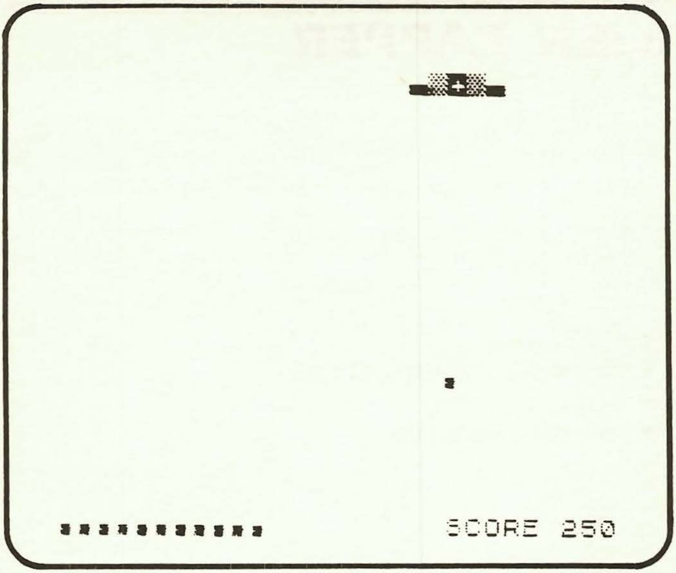
- The Game
- Program 3: ALIEN ZAPPER
- Points of Interest
- Possible Adaptations

The Game

In this game you are the last Earth defense base still operational, and you are feverishly warding off hordes of alien attackers swarming in at all levels of the atmosphere.

In order to eliminate aliens you must score a direct hit. Nipping the side bars doesn't count since it doesn't affect the oncoming alien. You are allowed 15 shots and firing is done at aliens located on four different levels. On the highest level, each successful shot is worth 1000 points; then in lower levels hits are worth 500, 250, and finally 100 points. If a hit is registered in the center of the alien, it is completely destroyed. However, if the light-shaded fuel tanks are hit, the alien ejects them to prevent explosions and it shrinks in size.

After each shot is fired, one unit of energy is expended and this is shown graphically on the lower left of the screen by the disappearance of one of the 15 small blocks. So keep your eye on that row of energy packages as you take aim. In order to fire at an alien located directly above, simply press any key on the keyboard. The game is over as soon as your reserves have been reduced to zero. Your final score is recorded for you at the bottom of the screen.



Program 3: ALIEN ZAPPER

```
100  LET D$="▄▒+▒▄"
110  LET S$="_ ▄+▄_ "
120  LET F$="_ . ▒. _ "
130  GOSUB 1000
140  LET PT=0
150  LET FLAG=0
160  FOR K=0 TO 14
170  IF FLAG< >0 THEN GOTO 200
180  GOSUB 2000
190  PRINT AT X,Y-2;D$
200  FOR I=0 TO 63
210  PLOT I,2
220  UNPLOT I,2
```

```
230  IF INKEY$<>"" THEN GOTO 260
240  NEXT I
250  GOTO 200
260  FOR J=2 TO 41
270  PLOT I,J
280  UNPLOT I,J
290  NEXT J
300  UNPLOT 28-2*K,0
310  GOSUB 3000
320  PRINT AT 21,20;"SCORE_";PT
330  NEXT K
340  GOTO 4000
1000 FOR K=0 TO 28 STEP 2
1010 PLOT K,0
1020 NEXT K
1030 RETURN
2000 LET X=INT (RND*4)+1
2010 LET Y=INT (RND*27)+3
2020 RETURN
3000 IF I=2*Y OR I=2*Y+1 THEN GOTO 3200
3005 IF FLAG=2 THEN RETURN
3010 IF I=2*Y-1 OR I=2*Y+2 OR I=2*Y-2 OR
     I=2*Y+3 THEN GOTO 3100
3020 LET FLAG=1
3030 RETURN
3100 PRINT AT X,Y-2;S$
3110 LET FLAG=2
3120 RETURN
3200 PRINT AT X,Y-2;F$
3210 PAUSE 50
```

```
3220  PRINT AT X,Y-1;" . . ."
3230  PAUSE 40
3240  PRINT AT X,Y-1;"_ _ _"
3250  LET Q1=(X=1)
3260  LET Q2=(X=2)
3270  LET Q3=(X=3)
3280  LET Q4=(X=4)
3290  LET PT=PT+(Q1*1000)+(Q2*500)
      +(Q3*250)+(Q4*100)
3300  LET FLAG=0
3310  RETURN
4000  FOR H=1 TO 10
4010  PRINT AT 20,11;"GAME_OVER"; AT 20,11;
      "GAME OVER"
4020  NEXT H
```

Points of Interest

Line 100: This is the string containing the graphics for a full-size alien.

Line 110: This string contains the graphics for a shrunken alien.

Line 120: This string contains the graphics for a partially disintegrated alien.

Lines 160–330: These lines contain the control for the random positioning of the aliens, the displaying of the radar blip, and also for checking whether an attempt has been made to fire at an alien.

Lines 1000–1030: This subroutine displays the 15 energy reserves.

Lines 2000–3030: This series checks whether a hit has been made.

Lines 3100–3120: These lines produce the image of a shrunken alien.

Lines 3200–3330: This group of lines takes care of the display of the disintegrating alien (in the event that a direct hit has been scored) and by means of Boolean logic calculates the appropriate score.

Lines 4000–4020: These lines flash the message "GAME OVER" and terminate the game.

Possible Adaptations

1. If the player wishes, the shape of the alien can be changed.
2. The allocation of points can be changed at will. The manner in which the points are calculated is based on the code:

   ```
   LET Q1 = (X=1)
   LET Q2 = (X=2)
   LET Q3 = (X=3)
   LET Q4 = (X=4)
   ```

   where X represents the level randomly selected in line 2000. Only one of the four expressions on the right of the equal sign can be true at any given time. Therefore, only one of the values Q1, Q2, Q3, and Q4 can have the value one, while all the rest must have the value 0. So all you have to do is to change the value in line 3290.

Chapter 4

REPEAT

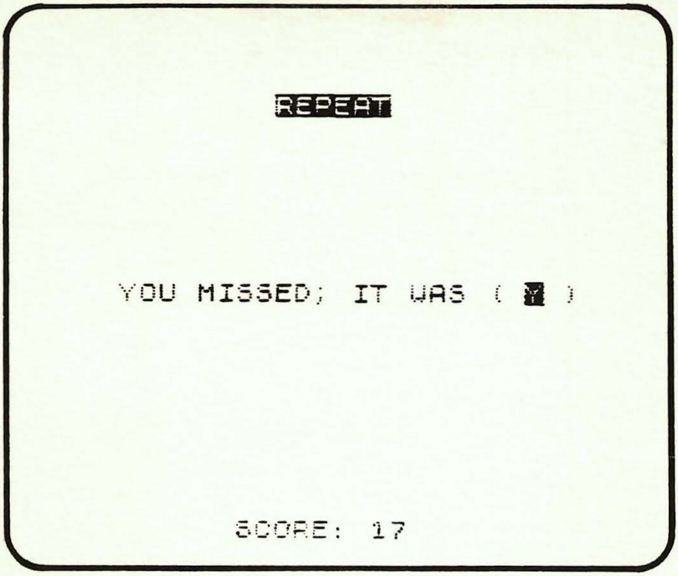
- The Game
- Program 4: REPEAT
- Points of Interest
- Possible Adaptations

The Game

The object of the game REPEAT is to type in the letter that appears in the middle of the screen as soon as possible after it has been displayed. If you are successful in typing the letter correctly in the amount of time allotted, a point is added to your score automatically. Your score is then displayed at the bottom of the screen, after which the next letter is presented. However, the time allowed for a response after each successive attempt is shortened. As soon as you either make a mistake or take more time than is allowed, the game is terminated, indicating which letter was missed, and what, if any, the total score was. The message "YOU GOT IT" is flashed in the event that you typed in the letter correctly, but the message "YOU MISSED IT" is displayed if it was either wrong or late. The pool of letters is taken from the top row of the keyboard.

Program 4: REPEAT

```
100  LET X$="QWERTYUIOP"
105  LET Y$="QWERTYUIOP"
110  LET S$="___________"
```




```
115  LET PT=Ø
12Ø  LET TIME=Ø
13Ø  FOR I=1 TO 1Ø
14Ø  PRINT AT 3,13;"REPEAT";AT 3,13;"REPEAT"
15Ø  NEXT I
155  LET Q=INT (RND*1Ø)+1
16Ø  LET Z$=X$(Q)
17Ø  LET TIME=TIME+1
18Ø  PRINT AT 11,15;Z$
19Ø  PAUSE 1ØØ-TIME
2ØØ  PRINT AT 11,15;"_"
21Ø  IF INKEY$< >Z$ THEN GOTO 3ØØ
22Ø  LET PT=PT+1
23Ø  PRINT AT 21,11;"SCORE:_";PT
24Ø  FOR I=1 TO 7
```

```
250  PRINT AT 11,10;"YOU_GOT_IT";AT 11,10;
     "YOU GOT IT";AT 11,10;S$
260  NEXT I
270  GOTO 155
300  PRINT AT 11,5;"YOU_MISSED;_IT_WAS_
     (_ _ _)"
310  FOR I=1 TO 10
320  PRINT AT 11,26;Z$;AT 11,26;Y$(Q)
330  NEXT I
```

Points of Interest

Line 100: In this line the letters QWERTYUIOP are stored in X$.

Line 105: In this line the inverse of the 10 letters of X$ is stored in Y$.

Line 110: A line of 10 spaces is stored in S$ for blanking out purposes.

Lines 155–270: This series randomly selects a character to flash, handles the timing of the response, checks to see if the response was correct, increments the score if it was correct and shortens the time allowed (**PAUSE** 100 — TIME).

Lines 300–330: If the response was incorrect, this segment prints the message that it was missed and displays the correct character.

Possible Adaptations

1. The **PAUSE** can always be altered to suit different participants.
2. The pool of letters used can always be changed for the sake of variety.

Chapter 5

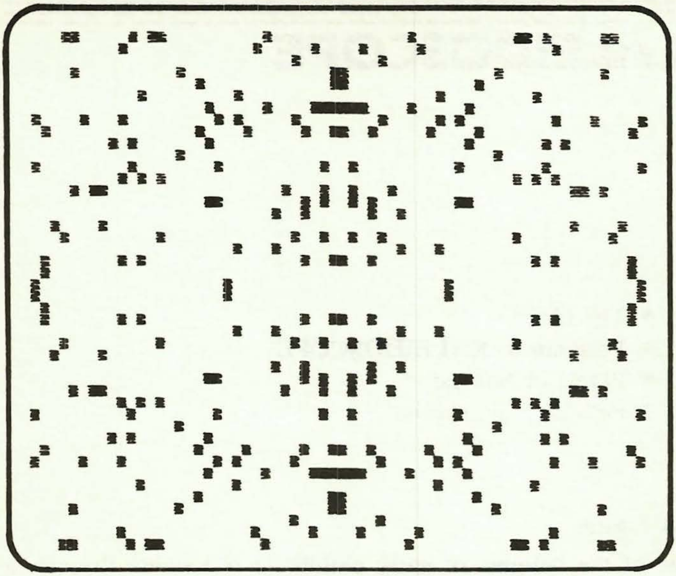
KALEIDOSCOPE

- The Game
- Program 5: KALEIDOSCOPE
- Points of Interest
- Possible Adaptations

The Game

One of the delights of early childhood is looking through a kaleidoscope. You will remember that when you shook the kaleidoscope, small pieces of mirrors changed their positions to create a startlingly vivid, symmetrical design. The slightest jolt of the cylinder changed the design dramatically, but it was always delightful to the eye and never ending in its variety.

To recreate those happy days, the next program uses the screen to simulate the kaleidoscope, using the **PLOT** and random functions to create random, symmetrical designs. The symmetry is assured by randomly selecting one of the four quadrants and reflecting each randomly selected pixel into each of the remaining quadrants. What is so nice about this program is that even if the same number is entered, the picture created is completely different. All you, the user, have to do is to type in the number of different designs you would like to see (and the complexity desired) and presto! your willing and able servant (the T/S computer) is at your service. Each design is displayed for a moment or so and after the screen has been cleared the next one is displayed. Once the requested number of designs has been displayed, the program comes to a normal termination. It is recommended that you select a complexity number between 50 and 100 for starters.



This is the first of our games which requires no thinking, no head scratching, no time limits and no frustration. Just sit back, relax, and enjoy!

Program 5: KALEIDOSCOPE

```
1ØØ  CLS
11Ø  PRINT AT 5,1Ø;"KALEIDOSCOPE"
12Ø  PRINT AT 12,2;"HOW_MANY_DESIGNS_DO_
     YOU_WANT_?"
13Ø  INPUT N
14Ø  PRINT AT 12,2;"HOW_COMPLEX_?_TYPE_
     ANY_NUMBER_"; AT 13,1Ø;"FROM_5Ø_TO_
     1ØØ"
15Ø  INPUT C
16Ø  FOR I=1 TO N
17Ø  CLS
```

```
180  FOR J=1 TO C
190  LET X=INT (RND*32)
200  LET Y=INT (RND*22)
210  PLOT X,Y
220  PLOT 63-X,Y
230  PLOT X,43-Y
240  PLOT 63-X,43-Y
250  NEXT J
260  PAUSE 60
270  NEXT I
```

Points of Interest

Lines 110–150: These lines allow the viewer to input the desired number of designs to be created.

Lines 160–270: This is the routine that generates the designs.

Possible Adaptations

1. The period of time allotted between each displayed design can be increased or decreased according to your personal wish. (Line 260 would have to be changed.)

PART II

Games for the Timex/Sinclair 1500

The twenty games in this part can be used on the T/S 1000 in conjunction with the 16K RAM pak, the T/S 1500, or the T/S 2068. None of these programs exceed 16K in length. Two of these games, Mastermind and Brain Stumper, cannot run on the T/S 2068, since they use the SCROLL command, which is a function that the T/S 2068 implements automatically and is not found on this model's keyboard. Some of these games use the FAST and SLOW commands. These commands are not needed on the T/S 2068; simply replace them with REM statements (followed by no REMark). Each of the T/S 2068 displays will be in black and white, since no color commands are included in any of the programs in this section.

Chapter 6

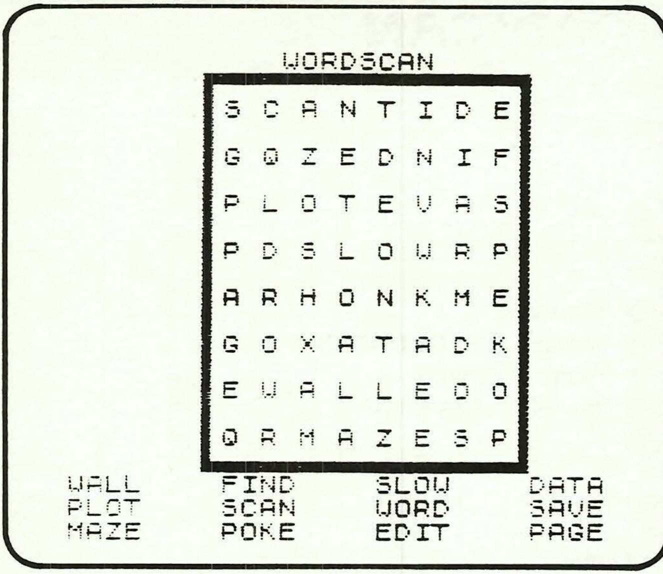
WORDSCAN

- The Game
- Program 6: WORDSCAN
- Points of Interest
- Possible Adaptations

The Game

We often see this type of game in daily newspapers where the reader is shown a square board crammed full of letters of the alphabet. Within this maze of letters (which looks randomly distributed) there are secreted some real words, usually geared to a particular topic or discipline. The object of the game is to find those words.

In this T/S version of the game, the board is 8 × 8 in dimension, meaning that there is room for 64 letters. There is a pool of 20 different four-letter words stored in a string and the player is asked to specify how many of these words are to be hidden. Once this has been specified, a special technique is utilized that simulates the READ/DATA statements found on other versions of BASIC but which is not available on the T/S version. The maximum number of words permitted is restricted to 12 because there is simply not enough room to allow any more. In fact, even selecting 12 sometimes causes problems because the computer tries to position the randomly selected words in random positions. The selected words may be displayed horizontally (left-to-right or right-to-left) or vertically (top-to-bottom or bottom-to-top). What's



more, the program has been designed so that two words may interlock if they share the same letter at the crossover point.

To make the task of finding the words even more challenging, all spaces remaining after the words have been placed in the 8 × 8 grid are filled with randomly selected letters of the alphabet. The randomly selected words are displayed below the grid on the screen, so there shouldn't be any cause for undue anxiety. However, if the player fails to detect the hidden letters and finds that his or her frustration level is rising just a little too rapidly, hitting any character, followed by the ENTER key, strips the board of all the redundant letters, and leaves only the hidden words in their randomly selected positions. In this way they can be viewed easily.

When the **RUN** command is issued and the player has entered the required number of words, the screen goes blank since the computer switches to **FAST** mode in order to do the considerable amount of computation that is necessary. As soon as that is done, the computer switches back to **SLOW** and the display mode for the rest of the game.

Program 6: WORDSCAN

```
100  DIM X$(8,8)
110  DIM Y$(48)
120  DIM A$(8,8)
130  DIM Z(20)
140  GOSUB 900
150  LET W$="HELPMAZEWORDFINDDATAEDITPAGE
     WALLSAVETESTLOADSLOWSCANPLOTPOKE
     RESTFILEBYTECASTPEEK"
160  LET F$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
170  FOR I=1 TO N
180  LET X=INT (RND*20)+1
190  IF Z(X)=1 THEN GOTO 180
200  LET Y$((I-1)*4+1 TO
     (I-1)*4+4)=W$((X-1)*4+1 TO (X-1)*4+4)
210  LET Z(X)=1
220  NEXT I
300  FOR I=1 TO N
310  LET Z$=Y$((I-1)*4+1 TO (I-1)*4+4)
320  LET X=INT (RND*8)+1
330  LET Y=INT (RND*8)+1
340  LET F1=0
350  IF X<5 AND Y<5 THEN GOSUB 3000
360  IF X<5 AND Y<5 AND F1=0 THEN GOSUB
     4000
370  IF X<5 AND Y>5 THEN GOSUB 5000
380  IF X<5 AND Y>5 AND F1=0 THEN GOSUB
     4000
390  IF X>5 AND Y<5 THEN GOSUB 3000
```

```
400  IF X>5 AND Y<5 AND F1=0 THEN GOSUB
     6000
410  IF X>5 AND Y>5 THEN GOSUB 5000
420  IF X>5 AND Y>5 AND F1=0 THEN GOSUB
     6000
430  IF F1=0 THEN GOTO 310
440  NEXT I
450  GOSUB 7000
460  GOSUB 1085
470  GOSUB 2000
500  FOR I=1 TO 8
510  FOR J=1 TO 8
520  IF X$(I,J)<>"_" THEN GOTO 540
530  LET X$(I,J)=F$(INT (RND*26+1))
540  NEXT J
550  NEXT I
560  GOTO 8000
900  CLS
910  PRINT AT 9,4;"HOW_MANY_WORDS_DO_
     YOU_WANT_?";AT 10,10;"1-12_ALLOWED"
920  INPUT N
930  FAST
1000 FOR I=1 TO 8
1010 FOR J=1 TO 8
1020 LET X$(I,J)="_"
1030 NEXT J
1040 NEXT I
1050 FOR I=1 TO 12
1060 LET Z(I)=0
```

```
1Ø8Ø  RETURN
1Ø85  SLOW
1Ø86  CLS
1Ø9Ø  PRINT AT 1, 11;"WORDSCAN"
11ØØ  PRINT AT 2,7;"▛▀▀▀▀▀▀▀▀▀▀▀
      ▀▀▀▀▜"
111Ø  FOR I=3 TO 17
112Ø  PRINT AT I,7;"▌";AT I, 23;"▐"
113Ø  NEXT I
114Ø  PRINT AT 18,7;"▙▄▄▄▄▄▄▄▄▄▄▄
      ▄▄▄▄▟"
2ØØØ  PRINT AT 19,Ø;"";
2Ø1Ø  FOR I=1 TO N
2Ø2Ø  PRINT Y$((I-1)*4+1 TO (I-1)*4+4);"_ _ _ _";
2Ø3Ø  NEXT I
2Ø4Ø  RETURN
3ØØØ  FOR K=Ø TO 3
3Ø2Ø  IF X$(X,Y+K)<>"_" AND X$(X,Y+K)
      <>Z$(K+1) THEN RETURN
3Ø3Ø  NEXT K
3Ø4Ø  FOR K=Ø TO 3
3Ø5Ø  LET X$(X,Y+K)=Z$(K+1)
3Ø6Ø  NEXT K
3Ø7Ø  LET F1=1
3Ø8Ø  RETURN
4ØØØ  FOR K=Ø TO 3
4Ø2Ø  IF X$(X+K,Y)<>"_" AND X$(X+K,Y)
      <>Z$(K+1) THEN RETURN
4Ø3Ø  NEXT K
```

```
4Ø4Ø  FOR K=Ø TO 3
4Ø5Ø  LET X$(X+K,Y)=Z$(K+1)
4Ø6Ø  NEXT K
4Ø7Ø  LET F1=1
4Ø8Ø  RETURN
5ØØØ  FOR K=Ø TO 3
5Ø2Ø  IF X$(X,Y-K)<>"_" AND X$(X,Y-K)
      <>Z$(K+1) THEN RETURN
5Ø3Ø  NEXT K
5Ø4Ø  FOR K=Ø TO 3
5Ø5Ø  LET X$(X,Y-K)=Z$(K+1)
5Ø6Ø  NEXT K
5Ø7Ø  LET F1=1
5Ø8Ø  RETURN
6ØØØ  FOR K=Ø TO 3
6Ø2Ø  IF X$(X-K,Y)<>"_" AND X$(X-K,Y)
      <>Z$(K+1) THEN RETURN
6Ø3Ø  NEXT K
6Ø4Ø  FOR K=Ø TO 3
6Ø5Ø  LET X$(X-K,Y)=Z$(K+1)
6Ø6Ø  NEXT K
6Ø7Ø  LET F1=1
6Ø8Ø  RETURN
7ØØØ  FOR I=1 TO 8
7Ø1Ø  FOR J=1 TO 8
7Ø2Ø  LET A$(I,J)=X$(I,J)
7Ø3Ø  NEXT J
7Ø4Ø  NEXT I
7Ø5Ø  RETURN
```

```
8ØØØ  FOR I=3 TO 17 STEP 2
8Ø1Ø  FOR J=8 TO 22 STEP 2
8Ø2Ø  PRINT AT I,J;X$(INT (I/2),INT ((J-5)/2))
8Ø3Ø  NEXT J
8Ø4Ø  NEXT I
8Ø5Ø  INPUT Q$
8Ø6Ø  FOR I=3 TO 17 STEP 2
8Ø7Ø  FOR J=8 TO 22 STEP 2
8Ø8Ø  PRINT AT I,J;A$(INT (I/2),INT ((J-5)/2))
8Ø9Ø  NEXT J
81ØØ  NEXT I
```

Points of Interest

Lines 170–220: These lines select the number of words requested from the string variable W$.

Lines 300–440: This series randomly locates the positions in which to place the word, determines the proper quadrant and sends the position to the appropriate subroutine to try to fit the word.

Line 450: This subroutine saves the answers in the matrix A$.

Lines 500–550: These lines insert random letters where blanks are found.

Lines 1085–2040: This series prints the game board and the selected words beneath the board.

Lines 3000–3080: These lines fit the words in the grid in a left-to-right direction.

Lines 4000–4080: These lines fit the words in a top-to-bottom direction.

Lines 5000–5080: These lines fit the words in a right-to-left direction.

Lines 6000–6080: These lines fit the words in a bottom-to-top direction.

Lines 8000–8100: This set prints the characters within the game board and, if desired, strips away the extra characters.

Possible Adaptations

1. The word list may be changed at the player's discretion. The simplest method is to replace some or all the words with other four-letter words.
2. The list of words could be extended to five-letter words but this would necessitate several other changes, including increasing the size of the board.
3. The player might want to try to include diagonals. This would, however, impose severe difficulties; but it could be done.
4. Of course, the randomly selected words shown at the bottom of the screen may be hidden from the player's view.
5. An innovative programmer might want to devise a scheme whereby, if the computer takes too long to place the words in the grid, an appropriate message is displayed and the player is asked to select another value for N. Even if the same value is selected it could happen that this time it will fit.

Chapter 7

ELECTRO-ZAP

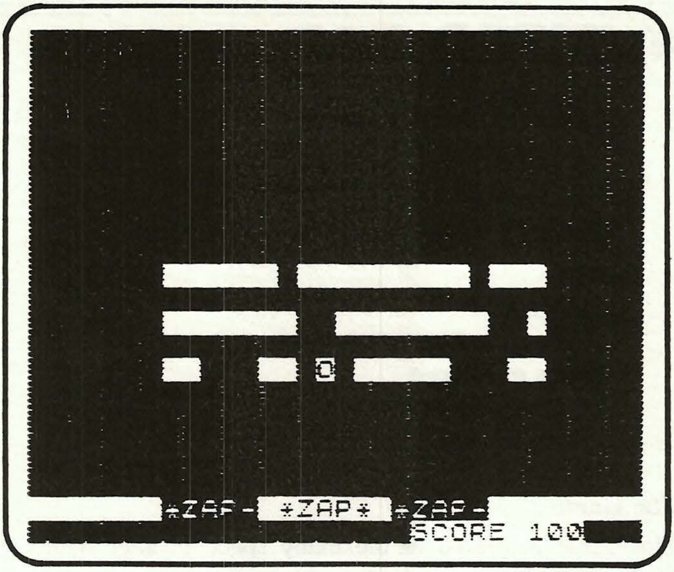
- The Game
- Program 7: ELECTRO-ZAP
- Points of Interest
- Possible Adaptations

The Game

This game is geared to the many space invader freaks out there. In the game you become a space spy whose mission is to somehow penetrate three rows of horizontal deadly light beams. If you hit one of these beams, or even if you linger too long over deciding what to do, you are zapped and the game is over.

On the screen, which incidentally, is the only time in this book that the entire game image is represented in inverse video, you are represented by the circle. When any key is pressed, the circle moves vertically upward (this is the only direction possible) while the electronic light beams continuously move in a horizontal direction, from right to left (for rows one and three) and left to right (for row two). You must try to penetrate the cracks in the beams without getting zapped. If you are (somehow) successful and reach the top of the screen, you are awarded points in increments of 100 and the speed of the light beams is increased, making your next foray even more difficult. To give you some heart, the number of points awarded is increased with each successful penetration. So go to it!

As will be seen from the length and complexity of the program which follows, this is not an easy game to program. It is equally difficult to achieve high scores.



Program 7: ELECTRO-ZAP

```
100  FAST
110  FOR I=Ø TO 21
120  FOR J=Ø TO 31
130  PRINT AT I,J;"■"
140  NEXT J
150  NEXT I
160  SLOW
170  LET A$="_ _ _ _ _ _ _ _ _ _ _■_ _ _ _ _ _ _
     _ _ _■"
180  LET B$="■■_ _ _ _ _ _ _ _ _ _■■_ _
     _ _ _ _ _ _ _"
190  LET C$="_ _ _ _ _ _■■■_ _ _ _■■■
     _ _■■■"
```

```
195  LET S$="________________"
200  LET PT=0
205  LET D1=0
210  LET ROW=20
220  LET FLAG=0
230  LET COL=15
240  PRINT AT ROW,COL;"O"
250  IF D1=10 THEN GOTO 280
260  LET D1=D1+1
270  GOTO 300
280  LET D1=1
300  FOR Z=1 TO 150
310  LET Y$=A$(1 TO D1)
320  LET A$(1 TO 20-D1)=A$(D1+1 TO 20)
330  LET A$(21-D1 TO )=Y$
340  LET T=10
350  GOSUB 1000
360  PRINT AT T,7;A$
370  IF INKEY$<>"" THEN GOSUB 1500
380  IF FLAG=1 THEN GOTO 210
400  LET Y$=B$(20)
410  LET B$(2 TO 20)=B$(1 TO 19)
420  LET B$(1)=Y$
430  LET T=12
440  GOSUB 1100
450  PRINT AT T,7;B$
460  IF INKEY$<>"" THEN GOSUB 1600
470  LET Y$=C$(1 TO D1)
480  LET C$(1 TO 20-D1)=C$(D1+1 TO 20)
```

```
490  LET C$(21-D1 TO )=Y$
500  LET T=14
510  GOSUB 1200
520  PRINT AT T,7;C$
530  IF INKEY$<>"" THEN GOSUB 1700
540  NEXT Z
550  GOTO 300
1000 IF ROW=10 AND A$(10)="_" THEN GOTO 2000
1010 RETURN
1100 IF ROW=12 AND B$(10)="_"THEN GOTO 2000
1110 RETURN
1200 IF ROW=14 AND C$(10)="_"THEN GOTO 2000
1210 RETURN
1500 IF ROW-1=10 AND A$(10)="_"THEN GOTO
     2000
1510 IF ROW>=8 THEN GOTO 1800
1520 PRINT AT ROW,COL;"■"
1530 LET PT=PT+D1*100
1540 LET FLAG=1
1550 PRINT AT 21,20;"SCORE_";PT
1560 RETURN
1600 IF ROW-1=12 AND B$(10)="_"THEN GOTO
     2000
1610 GOTO 1800
1700 IF ROW-1=14 AND C$(10)="_"THEN GOTO
     2000
1800 PRINT AT ROW,COL;"■"
1810 LET ROW=ROW-1
1820 PRINT AT ROW,COL;"O"
```

```
1830  RETURN
2000  PRINT AT ROW,COL;"█"
2010  FOR T=1 TO 10
2020  PRINT AT T,COL;"O";AT T,COL;"O"
2030  NEXT I
2040  FOR I=1 TO 15
2050  PRINT AT 20,0;S$;S$
2060  LET D$="*ZAP*"
2070  LET E$="*ZAP*"
2100  FOR I=1 TO 20
2110  PRINT AT 20,7;D$;TAB 13;E$;TAB 19;D$;AT
      20,7;E$;TAB 13;D$;TAB 19;E$
2120  NEXT I
```

Points of Interest

Line 100: This line puts the machine into **FAST** mode, during which time the screen goes blank. This is a good opportunity for you to gather strength for the formidable task ahead of you.

Lines 110–150: This series blacks out the entire printable area of the screen.

Line 160: This line restores the screen to **SLOW** and display mode.

Lines 170–190: These lines generate the three beams of light.

Lines 300–550: This section of coding handles the entire moving display together with the player's movements.

Lines 1000–1210: These three subroutines check whether you have stumbled into one of the light beams or if you have lingered too long in the darkened portions of the screen.

Lines 1500–1830: These lines determine if it is possible for you to occupy a particular location.

Lines 2000–2120: These lines display the flashing humanoid symbol, the circle in inverse video.

Possible Adaptations

1. The number of beams could be increased.
2. The speeds at which the beams move could be made to vary.
3. The direction of the beams could also be made to vary, perhaps randomly.

Chapter 8

HANGMAN

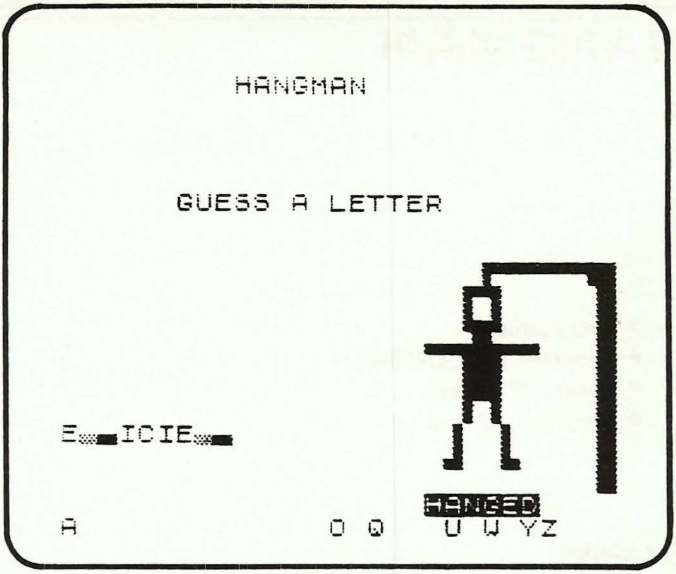
- The Game
- Program 8: HANGMAN
- Points of Interest
- Possible Adaptations

The Game

Who among us has not been totally captivated by the game of hangman sometime in our lives? Well, with the kindly aid of your precious T/S computer, you can recapture the delight that this game gave you right there on your screen. This version of the game responds to each incorrect answer by graphically building a scaffold and the body, piece by piece.

As you will recall, the game involves guessing the secret word in which each of the letters is represented by a blank space. At the beginning of the game all you know is the length of the word. By selecting letters of the alphabet that you think it might contain, you try to fill in the word within a certain number of tries. Sometimes the choices are rather obvious, but not always. In this version of the game the letters of the hidden word are represented by alternate bands of light and dark strips.

The pool is composed of 20 words of variable length; the maximum length is 10 letters. One of the 20 words is selected at random. If you select an incorrect letter, it is permanently displayed for you on the screen, in alphabetical order no less, to assist you in your future selections. If you do not succeed in guessing the word before the scaffold is built and you are



hanged, you lose the game (shame on you!) and you are greeted with the fact in inverse video. However, for each correct selection, the letter is automatically placed in its correct location (or locations, if it appears more than once in the word). If you succeed in completing the word, a message to that effect is displayed in inverse video and you may give yourself a hearty pat on the back.

Program 8: HANGMAN

```
100  LET X$ = "FINALIZE_ _FAMILIAR_ _
     COMMAND_ _ _ETERNITY_ _AVAILABLE
     _ADVANTAGE_REALISTIC_QUANTITY
     _ _CALCULATORFASCINATE_SUBTRACT
     _ _BENEATH_ _ _INFINITE_ _MATERIAL
     _ _SENSIBLE_ _EFFICIENT_ABSTRACT_
     _XYLOPHONE_SCIENTIST_PROGRAMMER"
```

```
11Ø  LET Y=INT (RND*2Ø) + 1
12Ø  LET W$=X$ ((Y-1)*1Ø+1 TO (Y-1)*1Ø+1Ø)
13Ø  LET H$=W$
14Ø  FOR I=1 TO 1Ø
15Ø  IF W$(I) = "_" THEN GOTO 19Ø
16Ø  NEXT I
17Ø  LET L=1Ø
18Ø  GOTO 2ØØ
19Ø  LET L=I-1
2ØØ  LET D$="▄▒▄▒▄▒▄▒▄▒"(1 TO L)
21Ø  PRINT AT 2,12, "HANGMAN"
22Ø  LET E=Ø
23Ø  PRINT AT 7,9;"GUESS_A_LETTER"
24Ø  PRINT AT 17,3;D$
25Ø  INPUT L$
26Ø  LET F=Ø
27Ø  FOR I=1 TO L
28Ø  IF L$=W$ (I) THEN GOTO 4ØØ
29Ø  NEXT I
3ØØ  IF F=1 THEN GOTO 23Ø
31Ø  LET E=E+1
32Ø  IF L$>="A" AND L$<="Z" THEN PRINT AT
     21, CODE (L$) -35;L$(1)
33Ø  GOTO E*1ØØ+5ØØ
4ØØ  LET D$(I)=W$(I)
41Ø  LET W$(I)="_"
42Ø  LET F=1
43Ø  IF D$=H$ (1 TO L) THEN GOTO 51Ø
44Ø  GOTO 27Ø
51Ø  FOR I=1 TO 1Ø
```

```
520  PRINT AT 2Ø,3;"YOU_GOT_IT
     ";AT 2Ø,3;"YOU GOT IT";AT 2Ø,3;"_ _ _ _
     _ _ _ _ _ _"
530  PRINT AT 17,3;"_ __ __ __ __ _"(1 TO L);
     AT 17,3;D$
540  NEXT I
550  STOP
600  PRINT AT 1Ø,25;"▛▀▀▀▀▜█"
610  FOR Q=11 TO 19
620  PRINT AT Q,31;"▐▌"
630  NEXT Q
640  GOTO 23Ø
700  PRINT AT 11,24;"▛▜"
710  PRINT AT 12,24;"▙▟"
720  GOTO 23Ø
800  PRINT AT 13,22;"▄▄▟"
810  GOTO 23Ø
900  PRINT AT 13,25;"▙▄▄"
910  GOTO 23Ø
1ØØØ PRINT AT 14,24;"██"
1Ø1Ø PRINT AT 15,24;"██"
1Ø2Ø GOTO 23Ø
11ØØ PRINT AT 16,24;"▌▐"
111Ø PRINT AT 17,23;"▐_ _▌"
112Ø GOTO 23Ø
12ØØ PRINT AT 18,23;"▟_ _▙"
121Ø FOR I=1 TO 1Ø
122Ø PRINT AT 2Ø,22;"HANGED";AT 2Ø,22; "HANGED"
123Ø NEXT I
```

Points of Interest

Line 100: This line stores the 20 words in the variable X$.

Lines 110–190: These lines randomly select one of the 20 words and strip it of blanks (if any are present).

Line 200: This line prints a line of alternate light and dark boxes to represent the length of the word selected by the computer.

Lines 230–420: This set allows the player to enter a letter; if it is present in the selected word, this segment of code places the letter in its proper place. If it is not a correct letter, a section of the scaffold is built via a calculated **GOTO** statement.

Lines 430–550: This series checks if the word is complete. If it is, it prints the message "YOU'VE GOT IT" in inverse video.

Lines 600–1230: This segment builds parts of the scaffold in succession; it also flashes the losing message "HANGED," again, in inverse video.

Possible Adaptations

1. You should feel free to change the words stored in X$. Should you do this, however, be sure to pad out to the right with blanks for each word less than 10 letters.
2. You might be tempted to increase the pool of words available.
3. A more challenging modification is to allow for the guessing of the whole word without having to type in the individual letters. Can you figure out how to accomplish this formidable task?
4. You may decide that printing the "already selected letters" is too much like spoon feeding, so you may want to eliminate this portion of the program.

Chapter 9

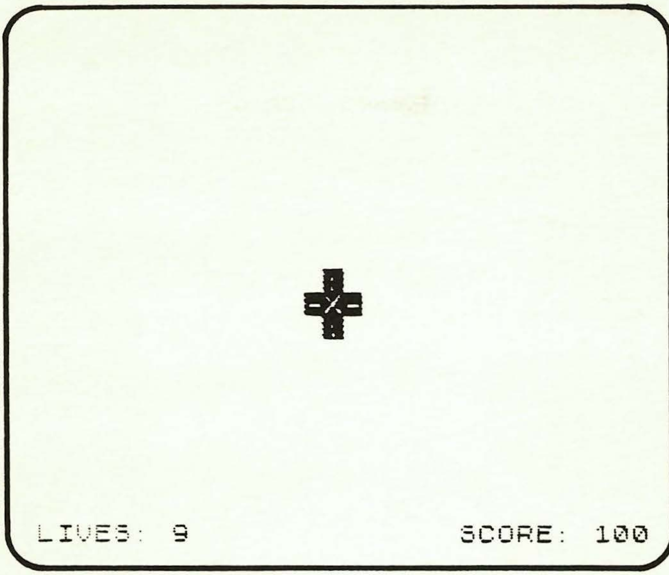
ORBIT

- The Game
- Program 9: ORBIT
- Points of Interest
- Possible Adaptations

The Game

Here is another space invader-type program. This time you are given a total of 10 lives. At the center of the screen is a space station. An alien invader is orbiting the station. If it makes one complete orbit without being hit, it lands and you lose one of your ten lives. Unfortunately, it is not visible all the time. In fact, it is visible only intermittently. Each time it shows its presence, it emits showers of deadly radiation. You can counter its moves by attacking it with the aid of four tractor beams. If the aim is successful, the tractor beam crushes the alien vessel. Even so, more aliens keep coming at you, keeping your hands full. For each one that you kill you are awarded 10 points. You can fire at them as often as you like. The four arrow keys located on the 5, 6, 7, and 8 keys are used to attack the aliens in the directions indicated on the keyboard. The game terminates when your life count reaches zero—which is inevitable.

The path taken by the alien vessel is always circular and it may start at any point on the periphery. A word of assistance in case you need it. Try your best to anticipate the position of the vessel before you actually take aim and fire. This game is a little too realistic for comfort!



Program 9: ORBIT

```
1ØØ  DIM Y(12,2)
1Ø5  LET X$="Ø115Ø32ØØ723112415
     231919211519111
     5Ø711Ø5Ø7Ø7Ø31Ø"
11Ø  LET P=Ø
115  LET L=1Ø
12Ø  FOR I=1 TO 12
13Ø  LET Q$=X$((I−1 *4+1 TO (I−1) *4+4)
14Ø  LET Y(I,1) =VAL Q$(1 TO 2)
15Ø  LET Y(I,2) =VAL Q$(3 TO 4)
16Ø  NEXT I
17Ø  GOSUB 1ØØØ
18Ø  LET S=INT (RND*6)+4
19Ø  LET FLAG=Ø
```

```
 200  FOR I=S TO 12
 210  PRINT AT Y(I,1),Y(I,2);"*"; AT Y(I,1),Y(I,2);"_"
 220  GOSUB 2000
 225  IF FLAG=1 THEN GOTO 350
 230  NEXT I
 240  FOR I=1 TO S
 250  PRINT AT Y(I,1),Y(I,2);"*";AT Y(I,1),Y(I,2);"_"
 260  GOSUB 2000
 265  IF FLAG=1 THEN GOTO 350
 270  NEXT I
 280  LET L=L-1
 290  FOR I=1 TO 5
 300  PRINT AT 21,13;"LANDED"; AT 21,13;
      "LANDED"; AT 21,13;"_ _ _ _ _ _"
 310  NEXT I
 350  PRINT AT 21,7;L;"_"; TAB (29);P
 360  IF L<>0 THEN GOTO 190
 370  FOR I=1 TO 10
 380  PRINT AT 21,11;"GAME_OVER"; AT 21,11;
      "GAME OVER"
 390  NEXT I
 400  STOP
1000  CLS
1010  PRINT AT 10,15;"▐"
1020  PRINT AT 11,14;"-×-"
1030  PRINT AT 12,15;"▐"
1040  PRINT AT 21,0;"LIVES:"; TAB (22);"SCORE:"
2000  IF INKEY$="5" THEN GOTO 2100
2010  IF INKEY$="6" THEN GOTO 2300
```

```
2Ø2Ø  IF INKEY$="7" THEN GOTO 25ØØ
2Ø3Ø  IF INKEY$="8" THEN GOTO 27ØØ
2Ø4Ø  RETURN
21ØØ  FOR K=13 TO 5 STEP -1
211Ø  PRINT AT 11,K;"▚";AT 11,K;"_"
212Ø  NEXT K
213Ø  IF Y(I,1)< >11 THEN RETURN
214Ø  GOTO 29ØØ
23ØØ  FOR K=13 TO 21
231Ø  PRINT AT K,15;"▚": AT K,15;"_"
232Ø  NEXT K
233Ø  IF Y(I,2)< >15 THEN RETURN
234Ø  GOTO 29ØØ
25ØØ  FOR K=9 TO 1 STEP -1
251Ø  PRINT AT K,15;"▚"; AT K,15;"_"
252Ø  NEXT K
253Ø  IF Y(I,2)< >15 THEN RETURN
254Ø  GOTO 29ØØ
27ØØ  FOR K=17 TO 24
271Ø  PRINT AT 11,K;"▚"; AT 11,K;"_"
272Ø  NEXT K
273Ø  IF Y(I,1)< >11 THEN RETURN
29ØØ  LET P=P+1Ø
291Ø  LET FLAG=1
292Ø  RETURN
```

Points of Interest

Line 100: This line dimensions the Y matrix for the circular path of the alien vessel.

Line 105: This line contains the coordinates for the orbit as string data.

Lines 120–160: These lines load the Y matrix with the coordinate data.

Lines 180–270: This series produces the display of the circular orbit.

Lines 280–360: These lines decrement the life count and produce the "LANDED" message in inverse video after the alien has successfully completed an orbit.

Lines 370–400: When the life count equals zero, the game is over and these lines display this message in inverse video.

Lines 2000–2040: These lines process the four possible direction keys.

Lines 2100–2920: This segment displays the activated tractor beams in whichever direction is used by the player. If the shot is successful, the score is incremented by 10.

Possible Adaptations

1. The number of points awarded for each hit can be raised or lowered to suit the maturity of the player.
2. The number of lives allowed can always be changed.
3. The speed with which the firing of the tractor beam is controlled can be changed, but this involves considerable work.
4. The path of the alien vessel can be changed by altering the string in line 105.

Chapter 10

SCRAMBLER

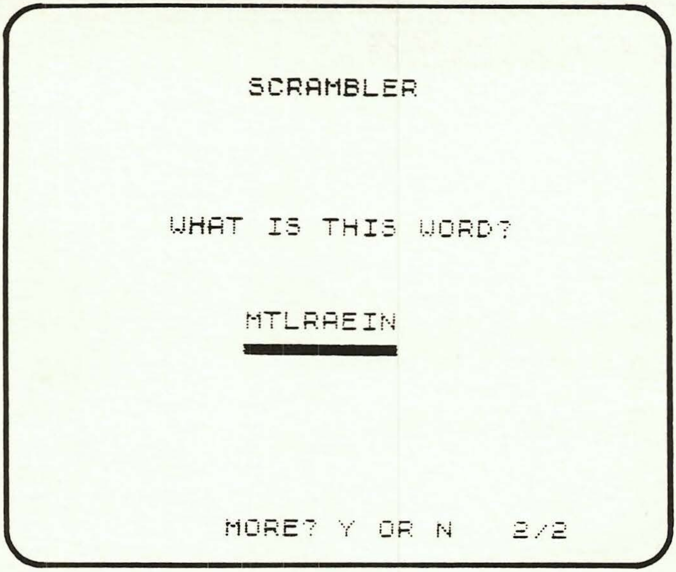
- The Game
- Program 10: SCRAMBLER
- Points of Interest
- Possible Adaptations

The Game

If the world-famous game of "Jumbles" turns you on, then this game is for you. The computer has stored in its memory 20 words of variable length, the maximum being 10 letters long. The computer randomly selects one of these words and jumbles up the letters, displaying them in the center of the screen and underlining them for your scrutiny. Your job is to unravel the original word. It is not always so easy, however practiced you might be. Just to add a little more excitement to the game, you are timed in your response.

Program 10: SCRAMBLER

```
100  LET X$="IMPOSSIBLETOUGH_ _ _ _ _
     MECHANIC_ _COMPUTER_ _PROCESS_ _ _
     TECHNICIANTRIVIAL_ _ _PHILOSOPHYEXOTIC
     _ _ _ _TERMINAL_ _NEUTRAL_ _ _TRANSFER
     _ _DIVERSE_ _ _CONVERSIONTELEPHONE_
     DIVISION_ _SCRAMBLE_ _MYSTERY_ _ _
     SIMPLE_ _ _ _KEYBOARD_ _"
```




```
105  LET S$="▄▄▄▄▄▄▄▄▄▄"
110  LET B$="_ _ _ _ _ _ _ _ _ _"
115  LET T=Ø
120  LET C=Ø
125  DIM A(1Ø)
130  DIM W$(1Ø)
140  CLS
150  PRINT AT 2,11;"SCRAMBLER"
155  LET W$=""
160  LET Y=INT (RND*2Ø)+1
170  LET T=T+1
180  LET T$=X$((Y-1)*1Ø+1 TO (Y-1)*1Ø+1Ø)
200  GOSUB 2ØØØ
210  PRINT AT 8,7;"WHAT_IS_THIS_WORD?"
220  LET Z=INT ((31-L)/2)
```

```
230  PRINT AT 12,Z;W$
240  PRINT AT 13,Z;S$(1 TO L)
250  PAUSE 2ØØ
260  IF INKEY$< >"" THEN GOTO 3ØØ
270  FOR I=1 TO 5
280  PRINT AT 21,9;"TIME_IS_UP"; AT 21,9;
     "TIME IS UP"; AT 21,9;B$
290  NEXT I
295  GOTO 4ØØ
300  INPUT R$
310  IF R$=H$ THEN GOTO 34Ø
320  PRINT AT 21,12;"WRONG"
330  GOTO 4ØØ
350  PRINT AT 21,11;"CORRECT"
360  LET C=C+1
400  PRINT AT 21,25;C;"/";T
410  PRINT AT 21,1Ø;"MORE?_Y_OR_N"
420  INPUT Q$
430  IF Q$="Y" THEN GOTO 14Ø
440  STOP
2ØØØ FOR I=1 TO 1Ø
2Ø1Ø IF T$(I)="_" THEN GOTO 2Ø5Ø
2Ø2Ø NEXT I
2Ø3Ø LET L=1Ø
2Ø4Ø GOTO 2Ø6Ø
2Ø5Ø LET L=I-1
2Ø6Ø LET H$=T$(1 TO L)
2Ø65 GOSUB 3ØØØ
2Ø7Ø FOR I=1 TO L
2Ø8Ø LET Q=INT (RND*L)+1
```

```
2Ø9Ø  IF A(Q)< >Ø THEN GOTO 2Ø8Ø
21ØØ  LET W$(I)=H$(Q)
21Ø5  LET A(Q)=1
211Ø  NEXT I
212Ø  IF W$=H$ THEN GOTO 2Ø65
213Ø  RETURN
3ØØØ  FOR I=1 TO L
3Ø1Ø  LET A(I)=Ø
3Ø2Ø  NEXT I
3Ø3Ø  RETURN
```

Points of Interest

Line 100: This line stores into the string variable X$, the 200 characters making up the pool of 20 words, 10 characters per word.

Lines 150–250: These lines print the header and the scrambled word which has been randomly selected from the pool, and begin the timing mechanism. Incidentally, the pause of 200 units is about three seconds.

Lines 260–290: This series flashes the message "TIME IS UP" in inverse video if the time limit has been exceeded.

Lines 300–440: This series accepts the player's answer, checks it, prints out whether the response is correct or not, increments the score counter if it is correct, and prompts the player whether another round is requested.

Lines 2000–2130: This group of lines strips the words of blanks (if any are present) and then proceeds to scramble the word.

Lines 3000–3030: These lines zero out the array A in preparation for the scrambling routine.

Possible Adaptations

1. Of course, once the words have become too familiar to the player they may be changed.
2. The time limit may be decreased or increased at will.

Chapter 11

BACCARAT II

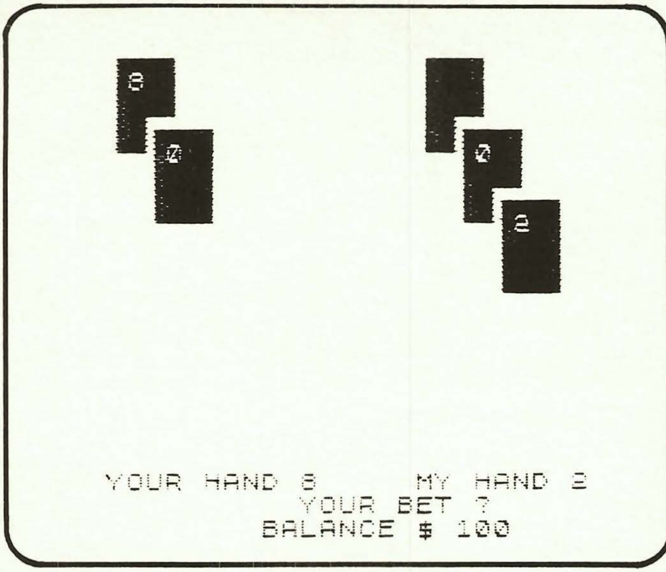
- The Game
- Program 11: BACCARAT II
- Points of Interest
- Possible Adaptations

The Game

The rules for this game are close to those found in your local casino. So, after a little practice with this game you should be ready to make a killing when you next visit the gambling halls. The player's hand is displayed on the left of the screen while the computer's hand is shown on the right. Two cards are dealt to each player and they must add up to 9 or as close to it as possible. All cards 10 or above count for 0, and any point value greater than 9, gets 10 subtracted from it. Therefore, a score of 15 is regarded as 5, and a score of 10 is considered to be 0.

Two cards are dealt both to the player and the computer. Both of the player's cards are dealt face up while one of the computer's cards is dealt face down. The player's point value is displayed while that of the computer is hidden from view. According to the way this game is programmed, the player always goes first.

Each card is shown in inverse video with the point value indicated in its top left hand corner. The player is prompted throughout the game for the appropriate response and a full betting routine is incorporated into the game. The messages: "TIE GAME," "YOU LOST," and "YOU WON" are



flashed on the screen when appropriate. The player starts out with a zero balance and may bet in any unit between $1 and $500. An entry of 0 terminates the game.

Program 11: BACCARAT II

```
100  LET X$="A 2 3 4 5 6 7 8 9 Ø J Q K"
1Ø5  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _"
11Ø  LET BAL=Ø
115  GOSUB 1ØØØ
12Ø  CLS
13Ø  LET R1=1
14Ø  LET C1=1
145  LET PT1=Ø
15Ø  FOR Q=1 TO 2
16Ø  GOSUB 2ØØØ
17Ø  GOSUB 3ØØØ
```

```
180  PRINT AT R1+1,C1+1;Y$
190  LET PT1=PT1+Y
200  LET R1=R1+3
210  LET C1=C1+2
220  NEXT Q
300  GOSUB 4000
310  PRINT AT 19,1;"YOUR_HAND_";PT1
400  LET R2=1
410  LET C2=17
420  LET PT2=0
430  FOR Q=1 TO 2
440  GOSUB 2500
450  GOSUB 3000
460  LET PT2=PT2+Y
470  IF Q=1 THEN GOTO 490
480  PRINT AT R2+1, C2+1;Y$
490  LET R2=R2+3
500  LET C2=C2+2
510  NEXT Q
600  GOSUB 4500
610  PRINT AT 19,17;"MY_HAND_■"
620  PRINT AT 17,1;"ANOTHER_CARD?"
630  INPUT A$
640  PRINT AT 17,1;S$
650  IF A$="N" THEN GOTO 750
660  GOSUB 2000
670  GOSUB 3000
680  PRINT AT R1+1,C1+1;Y$
690  LET PT1=PT1+Y
700  GOSUB 4000
```

```
710  PRINT AT 19,11;PT1
750  PRINT AT 19,25;PT2
760  IF PT2=PT1 THEN GOTO 850
770  IF PT2>PT1 THEN GOTO 900
780  IF Q=0 THEN GOTO 950
790  GOSUB 2500
800  GOSUB 3000
810  PRINT AT R2+1,C2+1;Y$
820  LET PT2=PT2+Y
830  GOSUB 4500
840  LET Q=0
845  GOTO 750
850  FOR I=1 TO 5
860  PRINT AT 20,12;"TIE_GAME"; AT 20,12;
     "TIE GAME"
870  NEXT I
875  LET W=0
880  GOTO 980
900  FOR I=1 TO 5
910  PRINT AT 20,12;"YOU_LOST"; AT 20,12;
     "YOU LOST"
920  NEXT I
930  LET W=-1
940  GOTO 980
950  FOR I=1 TO 5
960  PRINT AT 20,11;"YOU_WON"; AT 20,11;
     "YOU WON"
970  NEXT I
975  LET W=1
980  LET BAL=BAL+W*BET
```

```
990  PRINT AT 21,9;"BALANCE_$_":BAL;"_ __
     __"
995  GOTO 115
1ØØØ PRINT AT 2Ø,11;"YOUR_BET_?"
1Ø2Ø INPUT BET
1Ø3Ø PRINT AT 2Ø,11;S$
1Ø4Ø IF BET>=1 AND BET<=5ØØ THEN RETURN
1Ø5Ø IF BET=Ø THEN GOTO 5ØØØ
1Ø6Ø GOTO 1ØØØ
2ØØØ LET J=R1
2Ø1Ø LET K=C1
2Ø2Ø RETURN
25ØØ LET J=R2
251Ø LET K=C2
252Ø RETURN
3ØØØ PRINT AT J,K;"▗▄▄▖"
3Ø1Ø FOR G=J+1 TO J+3
3Ø2Ø PRINT AT G,K;"▐██▌"
3Ø3Ø NEXT G
3Ø4Ø PRINT AT J+4,K;"▝▀▀▘"
3Ø5Ø LET Y=INT (RND*13)+1
3Ø6Ø LET Y$=X$(Y)
3Ø7Ø IF Y>1Ø THEN LET Y=1Ø
3Ø8Ø RETURN
4ØØØ IF PT1<1Ø THEN RETURN
4Ø1Ø LET PT1=PT1-1Ø
4Ø2Ø GOTO 4ØØØ
45ØØ IF PT2<1Ø THEN RETURN
451Ø LET PT2=PT2-1Ø
452Ø GOTO 45ØØ
```

```
5ØØØ  FOR I=1 TO 1Ø
5Ø1Ø  PRINT AT 2Ø,11;"GAME OVER"; AT 20,11;
      "GAME OVER"
5Ø2Ø  NEXT I
```

Points of Interest

Line 100: This line stores the string of characters used for the card display, where the letter A stands for the ace, 0 for 10, J for jack, Q for queen, and K for king.

Lines 120–310: These lines print the player's cards and their values.

Lines 400–610: These lines handle the computer's cards, their values being hidden by a black box and one card face down.

Lines 620–710: This set asks the player if another card is required and processes the response accordingly.

Line 750: These lines reveal the value of the computer's cards.

Lines 760–995: This group of lines carries out the following chores: determines if another card is required, whether another game is requested, and what the balance is.

Lines 1000–1060: These lines incorporate the betting routine.

Lines 3000–3080: These lines display the card image.

Lines 4000–4020: These lines evaluate the player's hand.

Lines 4500–4520: These lines evaluate the computer's hand.

Lines 5000–5020: These lines flash the "GAME OVER" display in inverse video when appropriate.

Possible Adaptations

1. Instead of starting out the game with a balance of zero, the player could be given an arbitrary amount, for example, $100.
2. The betting limits could be changed to include further restrictions.

Chapter 12

MUNCHY

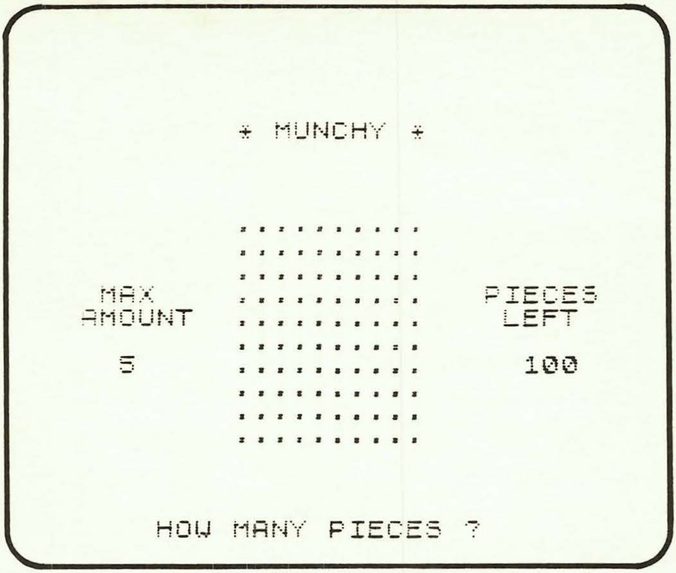
- The Game
- Program 12: MUNCHY
- Points of Interest
- Possible Adaptations

The Game

In this game you are situated in an orchard overflowing with ripe, juicy fruit. The orchard is represented by a 10 × 10 game board composed of 100 dots. The fruit has to be munched alternately between you and the computer. However, you have the choice, within limits, of how many pieces of fruit you can gobble up at one time. The idea of the game is not to be left with the last piece of fruit. The maximum number permitted varies between 5 and 9. It varies for each game in order to introduce an element of variety. All numeric responses are checked for validity. (Despite appearances, this game does not resort to matrices or arrays, but merely pointers.) The computer will always try to leave you with two more pieces of fruit than the maximum allowed in a fiendish endeavor to get you to lose. The words "CHOMP" and "MUNCH" are repeatedly displayed in inverse video for both the player and the computer.

Program 12: MUNCHY

```
1ØØ  LET T=1ØØ
1Ø5  LET P1=1
```




```
110  LET C1=100
115  LET D$="CHOMPCHOMP_ __ __MUNCH
     MUNCH"
120  LET Z$="*£:$+-?_"
130  LET M=INT (RND*5)+5
140  GOSUB 1000
150  PRINT AT 18,7;"HOW_MANY_PIECES_?"
160  INPUT N
170  PRINT AT 18,7;"_ __ __ __ __ __ __ __
     __"
180  IF N<T AND N>=1 AND N<=M AND N=INT
     (N) THEN GOTO 210
190  PRINT AT 18,8;"INVALID_NUMBER"
195  PAUSE 60
200  GOTO 150
```

```
210  LET T=T-N
220  LET P2=P1+(N-1)
230  GOSUB 2ØØØ
235  LET P1=P2+1
24Ø  GOSUB 3ØØØ
25Ø  PRINT AT 18,9;"WATCH_MY_MOVE"
3ØØ  IF T<=2*M+1 AND T>=M+3 THEN GOTO 38Ø
31Ø  IF T<=M+1 THEN GOTO 4ØØ
32Ø  IF T/M=INT (T/M) THEN GOTO 42Ø
33Ø  LET H=INT (T/M)
34Ø  LET G=T-H*M
35Ø  IF G=1 THEN GOTO 44Ø
36Ø  LET S=G-1
37Ø  GOTO 48Ø
38Ø  LET S=T-(M+2)
39Ø  GOTO 48Ø
4ØØ  LET S=T-1
41Ø  GOTO 48Ø
42Ø  LET S=M-1
43Ø  GOTO 48Ø
44Ø  IF H/2=INT (H/2) THEN GOTO 47Ø
45Ø  LET S=INT (RND*M)+1
46Ø  GOTO 48Ø
47Ø  LET S=M
48Ø  LET C2=C1-S+1
485  PRINT AT 19,8;"I_WANT_";S;"_PIECES"
49Ø  GOSUB 4000
495  T=T-S
5ØØ  LET C1=C2-1
```

```
505   PRINT AT 19,8;"_ __ __ __ __ __ _
      _ __ _"
510   GOSUB 5000
520   GOTO 150
1000  CLS
1010  PRINT AT 1,11;"*_MUNCHY_*"
1020  FOR I=5 TO 14
1030  FOR J=11 TO 20
1040  PRINT AT I,J;"."
1050  NEXT J
1060  NEXT I
1070  PRINT AT 8,4;"MAX";TAB (24);"PIECES"
1080  PRINT AT 9,3;"AMOUNT"; TAB (25);"LEFT"
1090  PRINT AT 11,5;M; TAB (26);T
1110  RETURN
2000  FOR I=P1 TO P2
2010  LET X=INT ((I-1)/10)+5
2020  LET Y=(I-INT (I/10)*10)+10
2030  IF Y=10 THEN LET Y=20
2040  PRINT AT 3,14;D$ (1 TO 5); AT 3,14;D$(6 TO 10);
      AT 3,14;D$(11 TO 15)
2050  PRINT AT 3,14;D$(1 TO 5); AT 3,14;D$(6 TO 10);
      AT 3,14;D$(11 TO 15)
2060  FOR K=1 TO 8
2070  PRINT AT X,Y;Z$(K)
2080  NEXT K
2090  NEXT I
2100  RETURN
3000  PRINT AT 11,26;T;"_ _"
3010  IF T<>1 THEN RETURN
```

```
3020  FOR I=1 TO 10
3030  PRINT AT 18,13;"YOU_WIN"; AT 18,13;
      "YOU WIN"
3040  NEXT I
3050  STOP
4000  FOR I=C1 TO C2 STEP -1
4010  LET X=INT ((I-1)/10)+5
4020  LET Y=(I-INT (I/10)*10)+10
4025  IF Y=10 THEN LET Y=20
4030  PRINT AT 16,14;D$(16 TO 20); AT 16,14;D$(21
      TO 25): AT 16,14;D$(11 TO 15)
4040  PRINT AT 16,14;D$(16 TO 20); AT 16,14;D$(21
      TO 25); AT 16,14;D$(11 TO 15)
4050  FOR K=1 TO 8
4060  PRINT AT X,Y;Z$(K)
4070  NEXT K
4080  NEXT I
4090  RETURN
5000  PRINT AT 11,26;T;"_ _"
5010  IF T< >1 THEN RETURN
5015  PRINT AT 18,9;"_ __ __ __ __ __ __ __"
5020  FOR I=1 TO 10
5030  PRINT AT 18,14;"I_WIN"; AT 18,14;"I WIN"
5040  NEXT I
```

Points of Interest

Line 115: This line contains the string D$ which holds the flashing inverse video message.

Line 120: This string, Z$, is used to give a chomping effect.

Lines 150–200: These lines test the value of the inputted N for validity.

Lines 210–240: These lines process the player's move.

Lines 250–520: This series handles the computer's move.

Lines 1000–1100: These lines set up the preliminary headers.

Lines 2000–2100: These lines physically display the player's move.

Lines 3000–3050: These lines flash the player's winning message.

Lines 4000–4050: These lines process and display the computer's move.

Lines 5000–5040: This set flashes the computer's winning message.

Possible Adaptations

1. The allowed maximum of chomps might be varied.
2. A time limit might be imposed to hurry the game along.

Chapter 13

SHADE IT IN

- The Game
- Program 13: SHADE IT IN
- Points of Interest
- Possible Adaptations

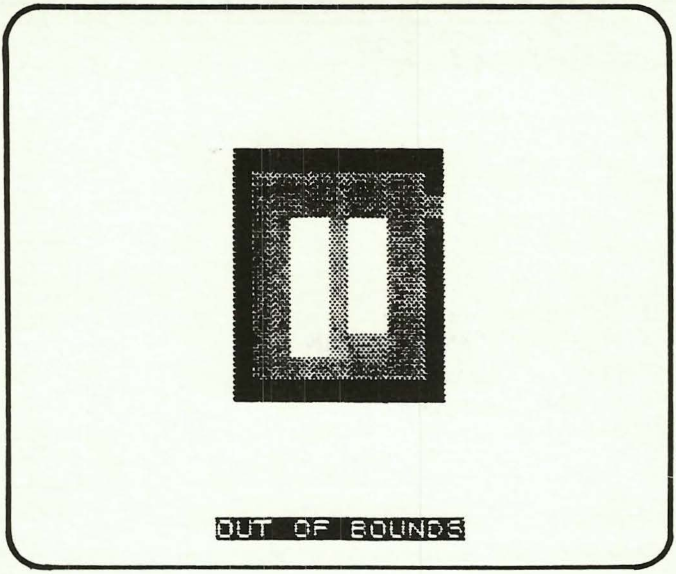
The Game

Regardless of how well you have fared until now, the next game is very likely going to trip you up. If you are the kind of person who never flinches when a severe challenge shows itself, start keying in the next program; but don't say you weren't warned of its difficulty!

The purpose of the game is to fill in the large, black box by controlling the passage of a light beam that snakes within the dark box. The large black box must be completely filled. In order to control the direction of the light beam, the keys 5, 6, 7, and 8 are used. They move the light beam in the directions indicated on the keys. Bear in mind that once a direction has been set, the light beam continues along its path until it is changed by the pressing of one of the direction keys. You are not allowed to hit any walls. If you do, an "OUT OF BOUNDS" message is flashed and you lose the game. Just in case this proves to be too easy, the game is further complicated by the fact that a time limit is imposed.

Program 13: SHADE IT IN

```
100  LET ROW=1
110  LET COL=5
```




```
120  DIM A(9,9)
130  LET TIME=0
135  LET Z=2
140  GOSUB 1000
150  GOSUB 2000
160  IF INKEY$<>"" THEN GOTO 200
170  GOSUB 3000
180  PAUSE 100
190  GOTO 160
200  LET Z$=INKEY$
210  IF Z$>="5" AND Z$<="8" THEN GOTO 230
220  GOTO 160
230  LET Z=VAL Z$
240  GOTO Z*100
500  LET COL=COL-1
```

```
510  IF COL<1 THEN GOTO 4ØØØ
52Ø  GOTO 9ØØ
6ØØ  LET ROW=ROW+1
61Ø  IF ROW>9 THEN GOTO 4ØØØ
62Ø  GOTO 9ØØ
7ØØ  LET ROW=ROW-1
71Ø  IF ROW<1 THEN GOTO 4ØØØ
72Ø  GOTO 9ØØ
8ØØ  LET COL=COL+1
81Ø  IF COL>9 THEN GOTO 4ØØØ
9ØØ  GOSUB 2ØØØ
91Ø  GOSUB 3ØØØ
92Ø  IF INKEY$<>"" THEN GOTO 2ØØ
93Ø  FOR I=1 TO 9
94Ø  FOR J=1 TO 9
945  IF A(I,J)=Ø AND Z=2 THEN GOTO 16Ø
95Ø  IF A(I,J)=Ø THEN GOTO Z*1ØØ
96Ø  NEXT J
97Ø  NEXT I
98Ø  FOR I=1 TO 1Ø
985  PRINT AT 2Ø,8;"YOU_FINISHED_IT"; AT 2Ø,8;
     "YOU FINISHED IT"
99Ø  NEXT I
995  STOP
1ØØØ LET F$="■■■■■■■■■■■"
1Ø1Ø PRINT AT 4,1Ø;F$; AT 14,1Ø;F$
1Ø2Ø FOR K=5 TO 13
1Ø3Ø PRINT AT K,1Ø;F$(1); TAB 2Ø;F$(1)
1Ø4Ø NEXT K
1Ø5Ø RETURN
```

```
2ØØØ  LET A(ROW,COL)=1
2Ø1Ø  LET X=ROW+4
2Ø2Ø  LET Y=COL+1Ø
2Ø3Ø  PRINT AT X,Y;"▒"
2Ø4Ø  RETURN
3ØØØ  LET TIME=TIME+1
3Ø1Ø  IF TIME>125 THEN GOTO 5ØØØ
3Ø2Ø  RETURN
4ØØØ  FOR I=1 TO 7
4Ø1Ø  PRINT AT ROW+4,COL+1Ø;"_";
      AT ROW+4,COL+1Ø; "▒"
4Ø2Ø  NEXT I
4Ø3Ø  FOR I=1 TO 1Ø
4Ø4Ø  PRINT AT 2Ø,9;"OUT_OF_BOUNDS"; AT 2Ø,9;
      "OUT OF BOUNDS"
4Ø5Ø  NEXT I
4Ø6Ø  STOP
5ØØØ  FOR I=1 TO 1Ø
5Ø1Ø  PRINT AT 2Ø,1Ø;"OUT_OF_TIME_";
      AT 2Ø,1Ø;"OUT OF TIME"
5Ø2Ø  NEXT I
```

Points of Interest

Lines 100–135: These lines set the initial values and dimension the matrix A.

Lines 160–240: These lines process the four directional keys, 5, 6, 7, and 8.

Lines 500–995: This group of lines moves the light beam in the desired direction and continues its passage along this path.

Lines 1000–1050: This is the subroutine to print the initial dark frame.

Lines 2000–2040: This set keeps track of the light beam.

Lines 4000–4060: These lines flash the display "OUT OF BOUNDS."

Lines 5000–5020: These lines flash the display "OUT OF TIME."

Possible Adaptations

1. The area covered by the dark frame could be enlarged for even greater fun.
2. The time limit may be adjusted.

Chapter 14

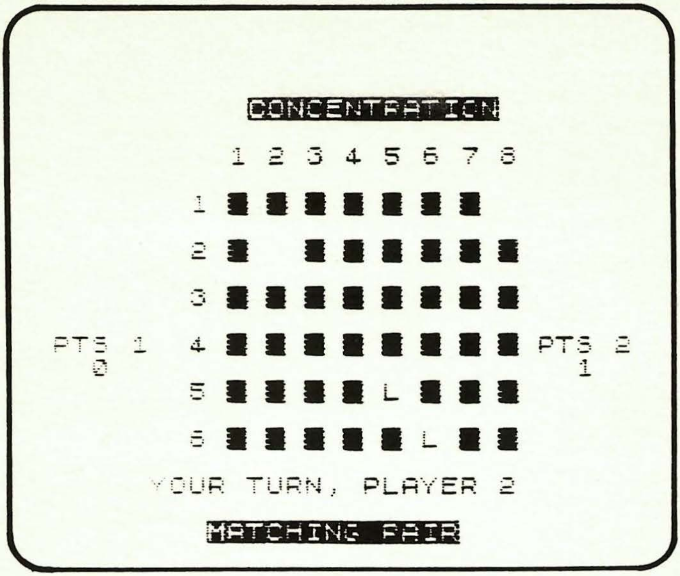
CONCENTRATION II

- The Game
- Program 14: CONCENTRATION II
- Points of Interest
- Possible Adaptations

The Game

The game of CONCENTRATION II is a computerized, T/S computer version of the popular card game called "Concentration." Rather than the 52 cards of the standard deck, this version of the game involves 48 electronic "cards," represented by the letters A through L, with four occurrences of each letter. (The suit in this game is of no consequence.) The computer randomly shuffles the cards with the computer in **FAST** mode before returning to the normal **SLOW** mode, at which time player number 1 goes first.

The forty-eight cards are represented by the 6 × 8 matrix A composed of 6 rows and 8 columns. A selection of a card is made by inputting the "coordinates" of the card—for example, row 3, column 5 would be entered by inputting the number 35. This is accepted as a two-character string and is stored in C$. (Care must be taken when entering the coordinates; a mistake is likely to trigger an OUT OF BOUNDS error.) Assuming the entry has been made correctly, the first card selected is revealed and the player is asked to make a second selection. Now, the second card is revealed. If they match, the player gains a point and continues, with the cards



being replaced by blanks that can never be accessed again. If the player makes a wrong second selection —in other words, if the two cards do not match—they are covered up again and the second player goes. The matrix is constantly on view and the points awarded are displayed to the left and right of the matrix; player 1's on the left and player 2's on the right. After each turn, the appropriate player is prompted to go by means of a polite message near the bottom of the screen.

The process is repeated for the second player. The game is terminated when all 24 pairs of cards have been matched. The winner is the player who has matched the most pairs of cards. The messages "CONCENTRATION," "NO MATCH,"and "MATCHING PAIR" are flashed on the screen when appropriate.

Program 14: CONCENTRATION II

```
100  DIM A(6,8)
110  FAST
115  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
```

```
120  LET PT1=Ø
130  LET PT2=Ø
140  LET X$="ABCDEFGHIJKL"
150  FOR N=1 TO 12
160  FOR M=1 TO 4
170  LET X=INT (RND*6)+1
180  LET Y=INT (RND*8)+1
190  IF A(X,Y)<>Ø THEN GOTO 17Ø
200  LET A(X,Y)=N
210  NEXT M
220  NEXT N
300  SLOW
310  CLS
320  FOR I=1 TO 5
330  PRINT AT 1,11;"CONCENTRATION";AT
     1,11;"CONCENTRATION"
340  NEXT I
350  PRINT AT 3,1Ø;"1_2_3_4_5_6_7_8"
360  FOR J=1 TO 6
370  PRINT AT 2*J+3,8;J;
380  FOR K=1 TO 8
390  PRINT "_█";
400  NEXT K
410  NEXT J
500  LET TURN=1
510  PRINT AT 17,6;"YOUR_TURN,_
     PLAYER_";TURN
515  LET FLAG=Ø
520  LET W$="1ST"
530  GOSUB 1ØØØ
```

```
540  GOSUB 2000
550  LET X1=X
560  LET Y1=Y
570  LET W$="2ND"
580  GOSUB 1000
590  GOSUB 2000
600  LET X2=X
610  LET Y2=Y
620  GOSUB 3000
630  PRINT AT 17,1;S$;S$
700  GOSUB 4000
710  IF TURN=1 THEN GOTO 750
720  GOTO 500
750  LET TURN=2
760  GOTO 510
1000 PRINT AT 20,8;"YOUR_";W$;"CARD_?"
1010 INPUT C$
1020 PRINT AT 20,8;S$
1030 IF C$(1)<"1" OR C$(2)<"1" OR C$(1)>"6" OR
     C$(2)>"8" THEN GOTO 1000
1040 LET X=VAL C$(1)
1050 LET Y=VAL C$(2)
1060 IF A(X,Y)=0 THEN GOTO 1000
1070 RETURN
2000 PRINT AT 2*X+3,(Y-1)*2+10;X$(A(X,Y))
2010 RETURN
3000 IF A(X1,Y1)=A(X2,Y2) THEN GOTO 3500
3010 FOR G=1 TO 5
3020 PRINT AT 19,11;"NO_MATCH";AT 19,11;
     "NO MATCH"
```

```
3Ø3Ø  NEXT G
3Ø4Ø  PRINT AT 19,11;S$
3Ø5Ø  LET P$="█"
3Ø6Ø  GOTO 36ØØ
35ØØ  FOR G=1 TO 5
351Ø  PRINT AT 19,9;"MATCHING_PAIR";AT 19,9;
      "MATCHING PAIR"
352Ø  NEXT G
353Ø  PRINT AT 19,9;S$
354Ø  IF TURN=1 THEN LET PT1=PT1+1
355Ø  IF TURN=2 THEN LET PT2=PT2+1
356Ø  LET P$="_"
357Ø  LET A(X1,Y1)=Ø
358Ø  LET A(X2,Y2)=Ø
359Ø  LET FLAG=1
36ØØ  PRINT AT 2*X1+3, (Y1-1)*2+1Ø;P$
361Ø  PRINT AT 2*X2+3, (Y2-1)*2+1Ø;P$
3615  IF FLAG=Ø THEN RETURN
362Ø  IF TURN=1 THEN GOTO 365Ø
363Ø  LET TURN=1
364Ø  RETURN
365Ø  LET TURN=2
366Ø  RETURN
4ØØØ  PRINT AT 11,1;"PTS_1";AT 12,3;PT1
4Ø1Ø  PRINT AT 11,26;"PTS_2";AT 12,28;PT2
4Ø2Ø  IF PT1+PT2<>24 THEN RETURN
4Ø3Ø  FOR G=1 TO 1Ø
4Ø4Ø  PRINT AT 21,11;"GAME_OVER";AT 21,11;
      "GAME OVER"
4Ø5Ø  NEXT G
```

Points of Interest

Line 100: This line reserves room for the matrix A.

Line 115: This is a line of 15 blanks stored in S$.

Lines 140–220: This series scrambles the deck of cards and hides them.

Line 300: This line returns the game to **SLOW** mode for playing ease.

Lines 330–410: These lines display the game board.

Lines 500–760: This series asks player for 1st and 2nd card and sends control to a subroutine to both display and compare them.

Lines 1000–1070: This is a subroutine to prompt player and to accept the card coordinates.

Lines 3000–3660: This series determines whether a match has occurred, displays the appropriate message and either blanks out the matching pair or covers up the mismatch. This segment also tallies the points and makes an entry into the matrix A so that the cards cannot be used again.

Lines 4000–4050: This segment displays both players' points (if any) and checks to see whether the game is over.

Possible Adaptations

1. The game could be amended to allow for more than two players.
2. Some error checking of the inputted coordinates could be incorporated into the program.

Chapter 15

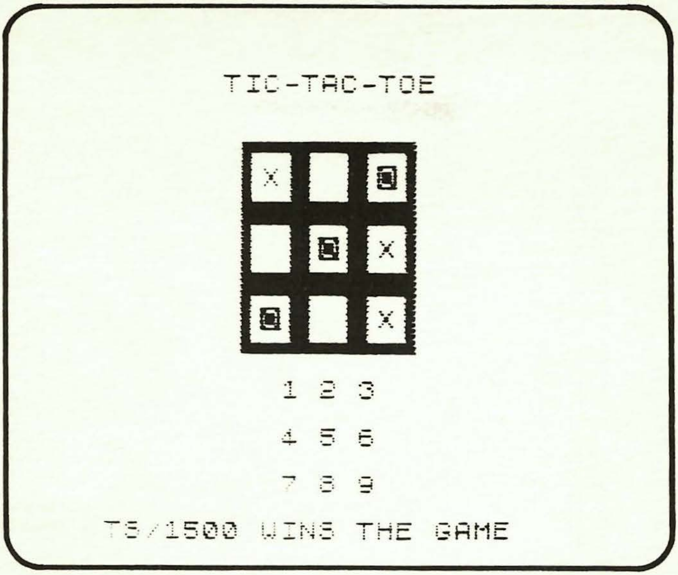
TIC-TAC-TOE

- The Game
- Program 15: TIC-TAC-TOE
- Points of Interest
- Possible Adaptations

The Game

This game hardly needs any description since it has been a universally acclaimed favorite for centuries by people of all ages. The game is built around a 3 × 3 set of boxes or squares into which a player puts either the letter O or the letter X, depending upon which of the two symbols has been selected for a particular player. In this specially devised version for the T/S, the player is matched against the computer and is given the symbol X, while the computer takes the symbol O. For ease of reference, the top row of boxes are labeled 1, 2, and 3, the second row 4, 5, and 6, and the bottom row 7, 8, and 9. The player is given the first opportunity to move. The idea is to place an X in three consecutive boxes, horizontally, vertically or diagonally. The computer tries to place an O in three consecutive boxes, at the same time blocking you from scoring. The game board is prominently displayed throughout the game, showing the symbol and the location of all selected moves by both you and the computer.

Now, it would appear that if the player is given the first move, he or she would have the advantage because the player has a total of five moves against the computer's four. Nevertheless, the program is so sophisticated in its design (as evidenced by its seemingly disproportionate length) that despite



many vigorous attempts, we have been unable to beat the computer. This is an open challenge to you!

The game is not timed, so you can be as careful and selective as you like before making a move. At the end of the game the winning sequence is displayed in inverse video. Our bet is that it will be the inverse **0** that is displayed, never the **X**. An appropriate message is displayed announcing the winner; but to be completely honest we have never been able to check out the segment of coding which displays "YOU BEAT THE T/S 1500" because it just hasn't happened, however hard we tried.

All numbers are validated for accuracy so that it is impossible to select an invalid box on the board. As we have said, you always have the privilege of going first and the winning sequence is displayed for you in inverse video.

Program 15: TIC-TAC-TOE

```
100  DIM D$(57)
110  LET K=0
```

```
120  LET T$= "▛▀▜▛▀▜▛▀▜"
130  LET M$="▌_▐▌_▐▌_▐"
140  LET B$= "▙▄▟▙▄▟▙▄▟"
150  LET D$="123456789147258369159357241689123
     721873896791632932715 93"
160  LET X$="_ _ _ _ _ _ _ _ _"
200  GOSUB 1000
210  PRINT AT 21,6;"WHICH_SPOT_?,_(1-9)"
220  INPUT S
230  PRINT AT 21,6;"_ _ _ _ _ _ _ _ _ _ _ _ _ _
     _ _ _ _ _"
240  IF S>=1 AND S<=9 THEN GOTO 280
250  PRINT AT 21,9;"INVALID_MOVE"
260  PAUSE 60
270  GOTO 210
280  IF X$(S)<>"_" THEN GOTO 250
300  LET K=K+1
310  LET X$(S)="X"
320  GOSUB 2000
330  GOSUB 3000
340  IF K=1 AND X$(5)="_" THEN LET X$(5)="O"
350  IF K=1 AND X$(5)="O" THEN GOTO 530
360  IF K=1 AND X$(5)="X" THEN LET X$(1)="O"
370  IF K=1 AND X$(1)="O" THEN GOTO 530
380  FOR I=1 TO 22 STEP 3
390  LET A=VAL D$(I)
400  LET B=VAL D$(I+1)
410  LET C=VAL D$(I+2)
420  IF X$(A)=X$(B) AND X$(A)<>"_" AND
     X$(C)="_" THEN GOTO 470
```

```
430  IF X$(A)=X$(C) AND X$(A)<>"_" AND
     X$(B)="_" THEN GOTO 490
440  IF X$(B)=X$(C) AND X$(B)<>"_" AND
     X$(A)="_" THEN GOTO 520
450  NEXT I
460  GOTO 600
470  LET X$(C)="O"
480  GOTO 530
490  LET X$(B)="O"
500  GOTO 530
520  LET X$(A)="O"
530  LET K=K+1
540  GOSUB 2000
550  GOSUB 3000
560  GOTO 210
600  GOSUB 4000
610  FOR I=Z TO 9
620  IF X$(I)<>"_" THEN GOTO 650
630  LET X$(I)="O"
640  GOTO 530
650  NEXT I
1000 CLS
1010 PRINT AT 2,10;"TIC-TAC-TOE"
1020 FOR I=5 TO 11 STEP 3
1030 PRINT AT I,11;T$
1040 PRINT AT I+1,11;M$
1050 PRINT AT I+2,11;B$
1060 NEXT I
1070 PRINT AT 15,13;"1_2_3"
1060 PRINT AT 17,13;"4_5_6"
```

```
1090  PRINT AT 19,13;"7_8_9"
2000  FOR I=1 TO 7 STEP 3
2010  PRINT AT I+5,12;X$(I);AT I+5,15;X$(I+1);AT
      I+5,18;X$(I+2)
2020  NEXT I
2030  RETURN
3000  FOR I=1 TO 22 STEP 3
3010  LET A=VAL D$(I)
3020  LET B=VAL D$(I+1)
3030  LET C=VAL D$(I+2)
3040  IF X$(A)=X$(B) AND X$(B)=X$(C) AND
      X$(C)<>"_" THEN GOTO 3100
3050  NEXT I
3060  IF K=9 THEN PRINT AT
      21,4;"NOBODY_WINS_THIS_ONE";G
3070  RETURN
3100  IF X$(A)="X" THEN GOTO 3200
3110  PRINT AT 21,4;"TS/1500_WINS_THE_GAME"
3120  LET X$(A)="O"
3130  LET X$(B)=X$(A)
3140  LET X$(C)=X$(A)
3150  GOSUB 2000
3160  PRINT G
3200  PRINT AT 21,4;"YOU_BEAT_THE_TS/1500"
3210  LET X$(A)="X"
3220  LET X$(B)=X$(A)
3230  LET X$(C)=X$(A)
3240  GOSUB 2000
3250  PRINT G
```

```
4ØØØ  IF K<>3 AND K<>5 THEN LET Z=1
4Ø1Ø  IF K<>3 AND K<>5 THEN RETURN
4Ø2Ø  FOR I=25 TO 55 STEP 3
4Ø3Ø  LET A=VAL D$(I)
4Ø4Ø  LET B=VAL D$(I+1)
4Ø5Ø  LET C=VAL D$(I+2)
4Ø6Ø  IF X$(A)=X$(B) AND X$(A)="X" THEN LET Z=C
4Ø7Ø  IF X$(A)=X$(B) AND X$(A)="X" THEN RETURN
4Ø8Ø  NEXT I
4Ø9Ø  LET Z=1
41ØØ  RETURN
```

Points of Interest

Lines 120–140: These lines set up the game grid for one row.

Line 150: All the possible moves the computer will make are listed here and are stored in the string D$.

Lines 210–270: All numeric input is validated by these lines.

Lines 340–370: This segment handles the very first move that the computer makes.

Lines 380–650: These lines read the string of numbers stored in D$ to determine whether searching is necessary. If not, they make the move.

Lines 1000–1090: This group of lines prints the game board and the numbers 1 through 9, representing the possible selections.

Lines 2000–2030: These lines print out the contents of the display.

Lines 3000–3070: These lines check to see if game is over.

Lines 3100–3250: This series determines who has won, the computer or the player.

Lines 4000–4100: These lines represent a special case of a trap that the computer must avoid.

Possible Adaptations

1. It might be desired to validate all inputted data, character and numeric.
2. Once numbers have been selected, they may be removed.
3. Perhaps you might want the winning line to be flashed.
4. Arrange for the computer to play against itself.
5. A timing arrangement could be implemented.

Chapter 16

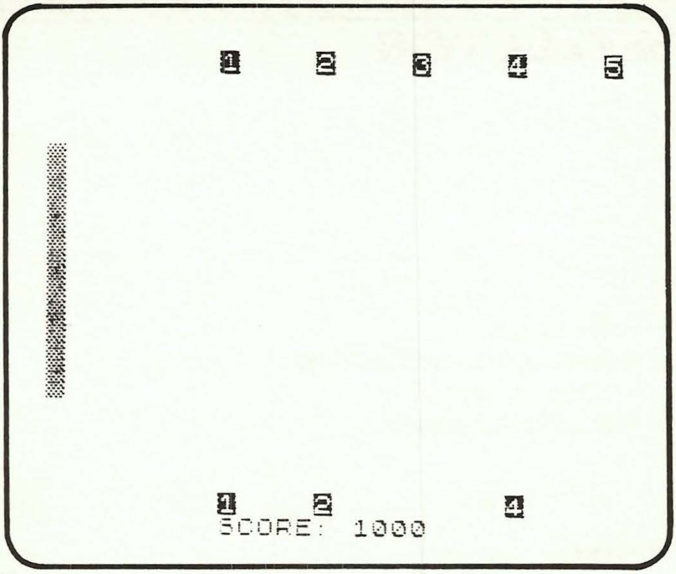
DEFLECTOR

- The Game
- Program 16: DEFLECTOR
- Points of Interest
- Possible Adaptations

The Game

In this fast moving game you are being attacked by high-powered photon balls being aimed at you from intergalactic attackers. You are in sole command of the five remaining earth stations. The attackers are unable to fire at you directly; they must fire at an orbiting satellite from which their shots are bounced off in your direction.

Whenever a photon ball is bounced off the satellite, it will head toward your five stations. It is then that you activate your defensive shields, with the proviso that at any given moment you can defend only one station. Keys 1, 2, 3, 4, and 5 are used to activate your shields. If you succeed, your station is left intact and you amass additional energy, represented by gaining some 500 extra points. If, on the other hand, you do not defend your station in time or you defend the wrong station, the station under attack will be demolished. In order for your complex shielding apparatus to work, you must keep at least two stations intact. For this reason, the game terminates when only one station remains. The game goes at a variable speed for increased fun. When a station is destroyed, the message "DESTROYED" is flashed; when the photon ball is deflected, the flashing message "DEFLECTED" is displayed.



So remember to get your fingers suitably positioned on the keyboard to activate the 1–5 keys.

Program 16: DEFLECTOR

```
100  DIM X(5)
110  LET Y$="12345"
120  LET S$="_ _ _ _"
125  LET B$="_ _ _ _ _ _ _ _ _ _ _"
130  GOSUB 1000
140  LET FIRE=INT (RND*5)+1
150  LET Z=VAL N$((FIRE-1)*4+1 TO (FIRE-1)*4+4)
160  LET Q=(FIRE-1)*5+10
170  LET SPD=INT (RND*2)+1
200  FOR I=2 TO 10 STEP SPD
210  PRINT AT I,Q;"O";AT I,Q;"O";AT I,Q;"_"
```

```
22Ø  LET Q=Q+Z*SPD
23Ø  NEXT I
24Ø  LET FLAG=Ø
3ØØ  LET FIRE=INT (RND*5)+1
31Ø  LET Z=VAL N$((FIRE-1)*4+1 TO (FIRE-1)*4+4)
32Ø  LET Q=2
33Ø  LET SPD=INT (RND*2)+1
34Ø  FOR I=11 TO 19 STEP SPD
35Ø  PRINT AT I,Q;"O";AT I,Q;"O";AT I,Q;"_"
36Ø  LET Q=Q-Z*SPD
37Ø  IF INKEY$<>"" THEN GOSUB 2ØØØ
373  PRINT AT 21,1Ø;"SCORE:_";PT
375  IF FLAG=1 THEN GOTO 14Ø
38Ø  NEXT I
39Ø  IF X(FIRE)=1 THEN GOTO 14Ø
4ØØ  FOR N=1 TO 7
41Ø  PRINT AT 2Ø,(FIRE-1)*5+1Ø;"▒";AT
     2Ø,(FIRE-1)*5+1Ø;"_"
42Ø  NEXT N
43Ø  FOR N=1 TO 7
44Ø  PRINT AT 21,1;"DESTROYED";AT 21,1;
     "DESTROYED"; AT 21,1;B$
45Ø  NEXT N
46Ø  LET HIT=HIT+1
47Ø  LET X(FIRE)=1
48Ø  IF HIT<4 THEN GOTO 14Ø
49Ø  FOR I=1 TO 1Ø
5ØØ  PRINT AT 12,11;"GAME_OVER";AT 12,11;
     "GAME OVER"
```

```
 51Ø   NEXT I
 52Ø   STOP
1ØØØ   FOR K=5 TO 15
1Ø1Ø   PRINT AT K,1;"▒"
1Ø2Ø   NEXT K
1Ø3Ø   PRINT AT 1,1Ø;
1Ø4Ø   FOR I=1 TO 5
1Ø5Ø   PRINT Y$(I);S$;
1Ø6Ø   NEXT I
1Ø7Ø   PRINT AT 2Ø,1Ø;
1Ø8Ø   FOR I=1 TO 5
1Ø9Ø   PRINT Y$(I);S$;
11ØØ   NEXT I
111Ø   LET PT=Ø
112Ø   LET HIT=Ø
113Ø   LET N$="-1.Ø-1.5-2.Ø-2.5-3.Ø"
114Ø   RETURN
2ØØØ   LET H$=INKEY$
2Ø1Ø   IF H$>="1" AND H$<="5" THEN GOTO 2Ø3Ø
2Ø2Ø   RETURN
2Ø3Ø   IF VAL H$=FIRE AND X(FIRE)=Ø THEN GOTO
       2Ø4Ø
2Ø35   RETURN
2Ø4Ø   LET FLAG=1
2Ø5Ø   FOR N=1 TO 7
2Ø6Ø   PRINT AT 21,23;"DEFLECTED";AT 21,23;
       "DEFLECTED";AT 21,23;B$
2Ø7Ø   NEXT N
2Ø8Ø   LET PT=PT+5ØØ
2Ø9Ø   RETURN
```

Points of Interest

Lines 140–230: These lines determine which of the five stations is to emit a photon ball and the speed with which it is emitted.

Lines 240–380: This set determines which of the five lower stations is targeted and at what rate of speed.

Line 390: This line checks to see whether the station has already been eliminated.

Lines 400–480: These lines eliminate one station and indicate that it has been destroyed by flashing the "DESTROYED" message in inverse video at left of the screen.

Lines 500–520: These lines flash the message "GAME OVER" and terminate the game.

Lines 1000–1140: This subroutine sets up the initial display and all initial variables.

Lines 2000–2090: This subroutine determines which of the five keys was struck and whether the deflection was successful.

Possible Adaptations

1. The number of points can be changed to suit the player's level.
2. The speed could be made constant, either very fast or very slow.

Chapter 17

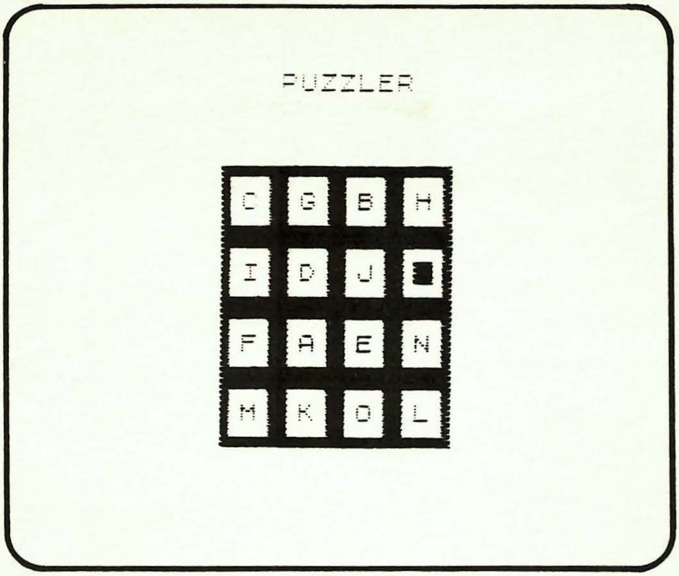
PUZZLER

- The Game
- Program 17: PUZZLER
- Points of Interest
- Possible Adaptations

The Game

This is a mosaic-type puzzle that is reminiscent of those sliding square puzzles that were the rage before the onslaught of Rubik's cube from Eastern Europe. The object of this game is to rearrange the 15 letters (A through O) and the one empty space that occupy a square composed of 16 boxes, into their correct alphabetical order. The space is provided in order to help you maneuver the letters around. The objective of the game is to rearrange the letters into their original order in the fewest number of moves.

There is no time limit in this game and it is just as well because, although the first few moves are relatively easy to make, after a little while the problem becomes extremely difficult, as you will see. The game opens up with the words: THIS IS YOUR OBJECTIVE. The final (winning) setting of the game board is then displayed for your edification—take a good look at it because it might be the last time you'll see the letters in their correct locations—and then after a short pause, the game goes into **FAST** mode during which time the screen blacks out and the shuffling of the lettered tiles takes place. After the shuffle has been completed, the game returns to **SLOW** mode and you are invited to make your first move.



When the computer prints out the message: WHICH PIECE?, you indicate your selection by stating which letter you want slid over. Often times, several different responses will have the same effect. If you want to quit (for any reason) just type in the letter X. If you (somehow) succeed in solving the mosaic puzzle, you are told how many moves it took you. You can use that figure as a benchmark for subsequent tries. Good luck!

Program 17: PUZZLER

```
1ØØ  DIM X$(4,4)
11Ø  LET Y$="ABCDEFGHIJKLMNO■"
12Ø  FOR I= 1 TO 4
13Ø  FOR J=1 TO 4
14Ø  LET X$(I,J) =Y$ ((I−1)*4+J)
15Ø  NEXT J
16Ø  NEXT I
```

```
165  LET C=Ø
17Ø  LET D=Ø
18Ø  LET F=Ø
19Ø  LET R=4
2ØØ  LET S=4
2Ø5  GOSUB 2ØØØ
21Ø  GOSUB 36ØØ
212  PRINT AT 2Ø,5;"THIS_IS_YOUR_OBJECTIVE"
213  PAUSE 6Ø
215  FAST
22Ø  FOR I=1 TO 1ØØ
23Ø  GOSUB 1ØØØ
24Ø  NEXT I
242  SLOW
243  GOSUB 2ØØØ
245  GOSUB 36ØØ
25Ø  PRINT AT 2Ø,1Ø;"WHICH_PIECE?"
26Ø  INPUT P$
265  PRINT AT 2Ø,1Ø;"_____________"
27Ø  IF P$="X" THEN GOTO 9ØØØ
28Ø  GOSUB 3ØØØ
29Ø  IF T=1Ø THEN GOTO 25Ø
3ØØ  FOR I=1 TO 4
31Ø  FOR J=1 TO 4
32Ø  IF X$(I,J)<>Y$((I-1)*4+J) THEN GOTO 36Ø
33Ø  NEXT J
34Ø  NEXT I
35Ø  GOTO 4ØØØ
36Ø  GOTO 25Ø
1ØØØ  LET W=R+INT (RND*3)-1
```

```
1Ø1Ø  LET Y=S+INT (RND*3)-1
1Ø2Ø  IF W>4 OR Y>4 OR W<1 OR Y<1 THEN
      GOTO 1ØØØ
1Ø3Ø  IF (W=R AND Y=S) OR (W< >R AND Y< >S)
      THEN GOTO 1ØØØ
1Ø4Ø  IF W=D AND Y=F THEN GOTO 1ØØØ
1Ø5Ø  LET X$(R,S)=X$(W,Y)
1Ø6Ø  LET X$(W,Y)="█"
1Ø7Ø  LET D=R
1Ø8Ø  LET F=S
1Ø9Ø  LET R=W
11ØØ  LET S=Y
111Ø  RETURN
2ØØØ  CLS
2Ø1Ø  PRINT AT 2,13;"PUZZLER"
2Ø2Ø  LET L$= "▛▀▜▛▀▜▛▀▜▛▀▜"
2Ø25  LET M$="▌_▐▌_▐▌_▐▌_▐"
2Ø3Ø  LET N$="▙▄▟▙▄▟▙▄▟▙▄▟"
2Ø4Ø  FOR I=6 TO 15 STEP 3
2Ø5Ø  PRINT AT I,1Ø;L$
2Ø6Ø  PRINT AT I+1,1Ø;M$
2Ø7Ø  PRINT AT I+2,1Ø;N$
2Ø8Ø  NEXT I
2Ø9Ø  RETURN
3ØØØ  FOR I=1 TO 4
3Ø1Ø  FOR J=1 TO 4
3Ø2Ø  IF P$< >X$(I,J) THEN GOTO 3Ø5Ø
3Ø3Ø  LET Z=I
3Ø4Ø  LET X=J
3Ø5Ø  IF X$(I,J)< >"█" THEN GOTO 3Ø8Ø
```

```
3Ø6Ø  LET N=I
3Ø7Ø  LET M=J
3Ø8Ø  NEXT J
3Ø9Ø  NEXT I
31ØØ  LET T=1Ø
311Ø  FOR Q=1 TO 3
312Ø  IF N=(Z+Q) AND M=X THEN LET T=Q-1
313Ø  IF N=(Z-Q) AND M=X THEN LET T=-(Q-1)
314Ø  IF M=(X+Q) AND N=Z THEN LET T=Q-1
315Ø  IF M=(X-Q) AND N=Z THEN LET T=-(Q-1)
316Ø  NEXT Q
317Ø  IF T=1Ø THEN RETURN
318Ø  IF ABS (T)+1<>1 THEN GOTO 321Ø
319Ø  LET X$(N,M)=X$(Z,X)
32ØØ  GOTO 35ØØ
321Ø  IF ABS (T)+1<>2 THEN GOTO 329Ø
322Ø  IF Z<>N THEN GOTO 326Ø
323Ø  LET X$(N,M)=X$(Z,X+T)
324Ø  LET X$(Z,X+T)=X$(Z,X)
325Ø  GOTO 35ØØ
326Ø  LET X$(N,M)=X$(Z+T,X)
327Ø  LET X$(Z+T,X)=X$(Z,X)
328Ø  GOTO 35ØØ
329Ø  IF Z<>N THEN GOTO 334Ø
33ØØ  LET X$(N,M)=X$(Z,X+T)
331Ø  LET X$(Z,X+T)=X$(Z,X+(T+(T/(-2))))
332Ø  LET X$(Z,X+(T+(T/(-2))))=X$(Z,X)
333Ø  GOTO 35ØØ
334Ø  LET X$(N,M)=X$(Z+T,X)
335Ø  LET X$(Z+T,X)=X$(Z+(T+(T/(-2))),X)
```

```
3360  LET X$(Z+(T+(T/(-2))),X)=X$(Z,X)
3500  LET X$(Z,X)="█"
3510  LET C=C+1
3600  FOR I=1 TO 4
3610  FOR J=1 TO 4
3620  PRINT AT (I-1)*3+7, (J-1)*3+11;X$(I,J)
3630  NEXT J
3640  NEXT I
3650  RETURN
4000  PRINT AT 21,3;"IT_IS_SOLVED_IN_";C;"_
      MOVES";B
9000  PRINT AT 20,10;"GAME_OVER"
```

Points of Interest

Lines 100–160: These lines set up the original board with the letters in their correct locations, described as your "objective."

Lines 250–360: This series accepts the input selection and validates it.

Lines 1000–1010: This is the scramble routine, and a thorough one it is too! It is a piece-by-piece scramble in which the blank space is exchanged with a neighbor no fewer than 100 times, checking that it does not reverse what has already been done, thereby ensuring that a solution is always possible.

Lines 3000–3650: These lines process the moves and manipulate the pieces.

Possible Adaptations

1. The character set may be changed at will by simply changing the contents of line 110.
2. A time limit could be imposed, if this is desired.
3. The number of moves allowed to a player could be limited.

4. The **FOR/NEXT** loop beginning in line 220 scrambles the letters, 100 times, which, of course, takes some definite amount of time. If this proves to be too long for comfort, the loop could be shortened to some lower figure.
5. Some scaled list of messages of congratulations could be implemented, depending upon how many tries were required to succeed.
6. Perhaps the **PAUSE** statement in line 213 could be changed to allow for more or less time for viewing the objective game board.

Chapter 18

SUPER SLOT

- The Game
- Program 18: SUPER SLOT
- Points of Interest
- Possible Adaptations

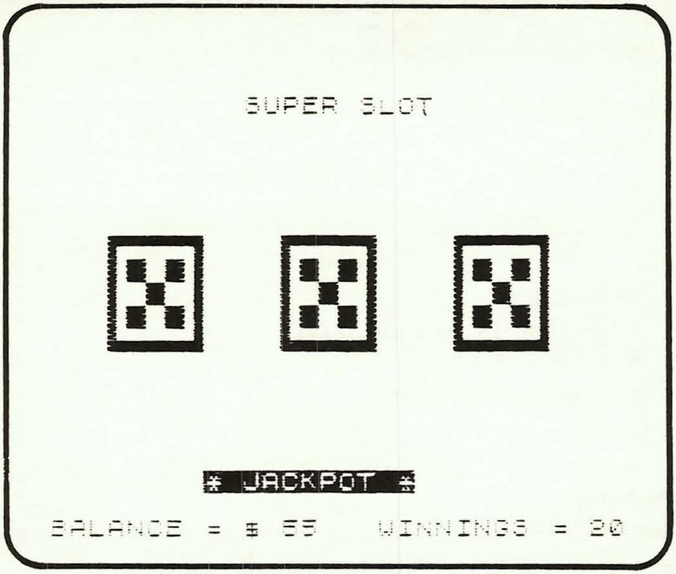
The Game

Most of us are familiar with the operation of slot machines, those so-called ''one-armed bandits'' that made Las Vegas so famous. This program simulates the spinning of the windows that are characteristic of the slot machine. The user is initially given an amount of $50 to start the gamble. He is allowed to make a bet of between $1 and $4 per shot. Each time a bet is made, a check is made to ensure that there is enough left in the balance to make the wager. If the test shows insufficient funds are available, the message: YOU DO NOT HAVE THAT MUCH is displayed. Any nonnumeric character is ignored. A bet of 0 terminates the game.

After you have made your bet, the game is set in motion. If you watch the screen carefully, you will see the three windows spin, one at a time, beginning with the leftmost. As soon as you hit any key on the keyboard, the first (leftmost) window stops. When you hit another key, the second window stops. Finally, hitting another key stops the third window.

Winning rounds are determined as follows:

- If all three windows display the same symbol, you have hit the proverbial jackpot, and the payout is five times the amount of the bet. A flashing message to this effect is then displayed.



- If the first two match, the payout is two times the amount of the original bet and a flashing message is displayed bearing the good tidings.
- Otherwise, you have lost the round and you are so advised by means of a flashing message.

The game always displays the balance and the winnings, if any. As soon as the balance has been reduced to zero the game is over and the flashing message: GAME OVER is displayed.

Program 18: SUPER SLOT

```
100  LET BAL=50
110  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
120  CLS
130  PRINT AT 3,11;"SUPER_SLOT"
140  LET R=8
```

```
150  LET C=4
160  GOSUB 1000
170  GOSUB 2000
180  LET W1=T
190  LET C=13
200  GOSUB 2000
210  LET W2=T
220  LET C=22
230  GOSUB 2000
300  IF W1=W2 AND W2=T THEN GOTO 400
310  IF W1=W2 THEN GOTO 450
320  FOR I=1 TO 5
330  PRINT AT 19,10;"_YOU_LOSE";
     AT 19,10;" YOU LOSE"
340  NEXT I
350  LET Z=-1
360  GOTO 700
400  FOR I=1 TO 7
410  PRINT AT 19,9;"*_JACKPOT_*"; AT 19,9;
     "* JACKPOT *"
420  NEXT I
430  LET Z=5
440  GOTO 700
450  FOR I=1 TO 5
460  PRINT AT 19,9;"*_2_MATCH_*"; AT 19,9;
     "* 2 MATCH *"
470  NEXT I
700  LET BAL=BAL+Z*B
710  PRINT AT 21,1;"BALANCE_=_$_";BAL;S$
```

```
715   IF BAL<=Ø THEN GOTO 3ØØØ
72Ø   IF Z>Ø THEN PRINT AT 21,18;
      "WINNINGS_=_";B*Z
73Ø   PAUSE 1ØØ
74Ø   PRINT AT 21,18;S$
745   PRINT AT 19,9;S$
75Ø   FOR I=9 TO 13
76Ø   PRINT AT I,4;S$;S$
77Ø   NEXT I
78Ø   GOTO 14Ø
1ØØØ  PRINT AT 2Ø,11;"BET_?_1-4"
1Ø1Ø  INPUT B
1Ø2Ø  PRINT AT 2Ø,11;S$
1Ø3Ø  IF B>=1 AND B<=4 AND B<=BAL
      THEN RETURN
1Ø4Ø  IF B=Ø THEN GOTO 3ØØØ
1Ø5Ø  IF B>BAL THEN GOTO 1Ø7Ø
1Ø6Ø  GOTO 1ØØØ
1Ø7Ø  PRINT AT 2Ø,3;"YOU_DO_NOT_HAVE
      _THAT_MUCH"
1Ø8Ø  PAUSE 6Ø
1Ø9Ø  PRINT AT 2Ø,3;S$;S$
11ØØ  GOTO 1ØØØ
2ØØØ  PRINT AT R+1,C;"▛▀▀▀▜"
2Ø1Ø  PRINT AT R+2,C;"▌█_█▐"
2Ø2Ø  PRINT AT R+3,C;"▌_█_▐"
2Ø3Ø  PRINT AT R+4,C;"▌█_█▐"
2Ø4Ø  PRINT AT R+5,C;"▙▄▄▄▟"
2Ø5Ø  LET T=1
2Ø6Ø  IF INKEY$<>"" THEN RETURN
```

```
21ØØ  PRINT AT R+1,C;"█████"
211Ø  PRINT AT R+2,C;"█_█_█"
212Ø  PRINT AT R+3,C;"██_██"
213Ø  PRINT AT R+4,C;"█_█_█"
214Ø  PRINT AT R+5,C;"█████"
215Ø  LET T=2
216Ø  IF INKEY$<>"" THEN RETURN
22ØØ  PRINT AT R+1,C;"█▀▀▀█"
221Ø  PRINT AT R+2,C;"▌▚_▞▐"
222Ø  PRINT AT R+3,C;"▌_█_▐"
223Ø  PRINT AT R+4,C;"▌▞_▚▐"
224Ø  PRINT AT R+5,C;"█▄▄▄█"
225Ø  LET T=3
226Ø  IF INKEY$<>"" THEN RETURN
227Ø  GOTO 2ØØØ
3ØØØ  FOR I=1 TO 1Ø
3Ø1Ø  PRINT AT 2Ø,11;"GAME_OVER"; AT 2Ø,11;
      "GAME OVER"
3Ø2Ø  NEXT I
```

Points of Interest

Lines 100–130: These lines initialize the variables.

Lines 140–230: Control section for the displaying of the three windows.

Lines 300–470: This series determines a win or loss.

Lines 700–780: These lines adjust the bounds to a win or loss.

Lines 1000–1100: This subroutine accepts and validates the bet.

Lines 2000–2060: These lines display the first of the three symbols.

Lines 2100–2160: These lines display the second of the three symbols.

Lines 2200–2270: These lines display the third of the three symbols.

Lines 3000–3020: When the balance falls to zero, these lines indicate the end of the game.

Possible Adaptations

1. The starting amount could always be changed. You might want the game to last a long time. In that case, you could make the starting amount something extremely large, like $1000 or more.
2. The limitation of $1 to $4 per bet can also be changed at will.
3. If you are in an artistic mood and are willing to invest the time required, you could change the symbols to represent the three windows.

Chapter 19

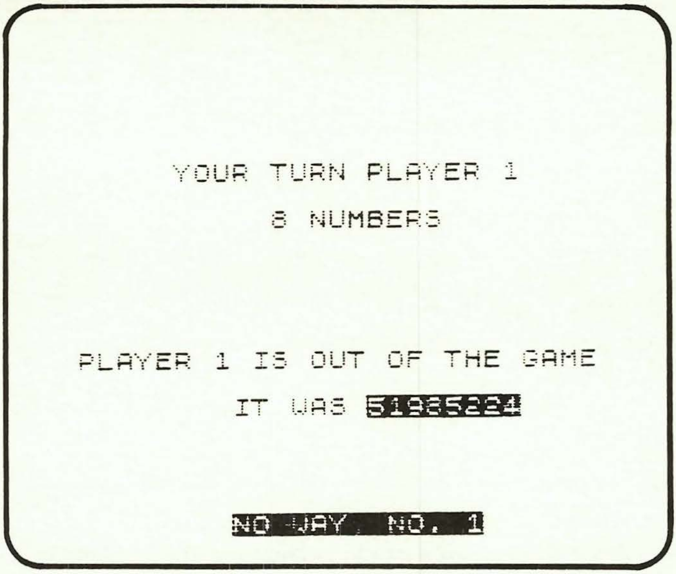
NUMBERS UP

- The Game
- Program 19: NUMBERS UP
- Points of Interest
- Possible Adaptations

The Game

This game is a variation of the popular electronic game called "Simon." Up to five players may participate, and with each round the difficulty is increased. The idea behind the game is this: The computer displays in the middle of the screen a random sequence of numbers, one at a time. You have to repeat those numbers in the exact sequence in which they were displayed on the screen. Any player who fails is immediately eliminated from the game, and for his express benefit, the correct sequence is displayed. To commence the game, any key on the keyboard is pressed. In this way each player has ample opportunity to position himself for optimum comfort.

The message: GET READY PLAYER NO ... (with a number 1 to 5 following NO), for each player is displayed after the computer is told how many players are in the game, and the game is in session. It is only fair to say that the game seems fairly easy at the beginning. However, as the game proceeds it gets progressively more difficult. If you are fortunate enough to get the sequence right, the computer flashes the message: YOU GOT IT, NO ... However, should the wrong sequence be entered, the computer advises you of this with the flashing message: NO WAY, NO ..., and the screen announces that that particular player is no longer in the game.



The following round includes all the remaining players, but the number of digits displayed is increased by one. But that is not all; the flashing of the numbers is speeded up. In fact, it would not be an exaggeration to say that they are flashed very fast indeed. The maximum sequence possible contains 10 numbers, but it is safe to bet that you will never reach it. (Yes, that's a challenge!) Once all the players have been eliminated, the message "GAME OVER" is flashed and the score for each player is listed.

After the player succeeds in repeating the sequence correctly, the flashing message: YOU GOT IT, NO ..., is displayed and the successful participant is identified by number. The chances are that after you have had an opportunity to become acquainted with this game, you will probably come to the conclusion that it should have been called: NOT SO SIMPLE SIMON!

Program 19: NUMBERS UP

```
100  LET X$="0123456789"
110  LET Y$="0123456789"
```

```
120  DIM D$(10)
130  DIM F$(10)
140  DIM P(5)
150  DIM S(5)
160  LET T1=4
170  PRINT AT 10,7;"HOW_MANY_PLAYERS_?";
     AT 11,9;"1_-_5_CAN_PLAY"
180  INPUT NUM
190  CLS
200  IF NUM<=0 OR NUM>5 THEN GOTO 170
210  LET W$="GET_READY_PLAYER_NO._"
220  LET V$="GET READY PLAYER NO.  "
230  LET S$="________________"
300  FOR K=1 TO NUM
310  IF P(K)=1 THEN GOTO 510
320  PRINT AT 3,7;"YOUR_TURN_PLAYER_";K
330  PRINT AT 5,12;T1;"_NUMBERS"
340  GOSUB 1000
350  PRINT AT 21,3;"HIT_ENTER_KEY
     _WHEN_READY"
360  IF INKEY$=" " THEN GOTO 360
370  PRINT AT 21,1;S$;S$
380  FOR J=1 TO T1
390  LET Z=INT (RND*10)+1
400  LET D$(J)=X$(Z)
410  LET F$(J)=Y$(Z)
420  PRINT AT 11,15;D$(J)
430  FOR Q=1 TO (10-T1)
440  NEXT Q
450  PRINT AT 11,15;"_"
```

```
460  NEXT J
470  PRINT AT 16,3;"REPEAT_THE_NUMBERS
     _PLAYER_";K
480  INPUT R$
485  PRINT AT 16,1;S$;S$
490  IF R$=D$ (1 TO T1) THEN GOSUB 2000
500  IF R$< >D$(1 TO T1) THEN GOSUB 3000
510  NEXT K
550  IF T1<10 THEN LET T1=T1+1
560  GOSUB 4000
570  GOTO 300
1000 FOR G=1 TO 7
1010 PRINT AT 21,4;W$;K; AT 21,4;V$;Y$(K+1)
1020 NEXT G
1030 RETURN
2000 FOR G=1 TO 5
2010 PRINT AT 18,8;"YOU_GOT_IT,_NO._";
     K; AT 18,8;"YOU GOT IT, NO.   ";Y$(K+1)
2020 NEXT G
2030 PRINT AT 18,1;S$;S$
2040 LET S(K)=S(K)+1
2050 RETURN
3000 FOR G=1 TO 5
3010 PRINT AT 18,10;"NO_WAY,_NO._";K; AT
     18,10;"NO WAY, NO.   ";Y$(K+1)
3020 NEXT G
3030 PRINT AT 11,2;"PLAYER_";K;"_IS_OUT_OF_
     THE_GAME"
3040 PRINT AT 13,10;"IT_WAS_"
```

```
3050  FOR G=1 TO 12
3060  PRINT AT 13,17;D$(1 TO T1); AT 13,17;F$
      (1 TO T1)
3070  NEXT G
3080  LET P(K)=1
3090  PRINT AT 18,10;S$; AT 11,1;S$;S$; AT 13,1;S$;S$
3100  RETURN
4000  FOR G=1 TO NUM
4010  IF P(G)=0 THEN RETURN
4020  NEXT G
4030  CLS
4040  FOR I=1 TO 10
4050  PRINT AT 3,11;"GAME_OVER"; AT 3,11;
      "GAME OVER"
4060  NEXT I
4070  PRINT AT 21,6;"ALL_PLAYERS_ARE_OUT"
4080  PRINT AT 7,9;"FINAL_SCORES"
4090  FOR R=10 TO (10+(NUM-1))
4100  PRINT AT R,7;"PLAYER  "; Y$(R-8);
      "  ===>  _ _";S(R-9)
4110  NEXT R
```

Points of Interest

Line 100: A string containing the ten possible digits 0 through 9.

Line 110: A string of the inverse digits 0–9 to create the flashing effect.

Lines 170–200: These lines allow for the option of one to five players to participate.

Lines 210–220: These lines set up the preparatory inverse message.

Line 230: A line of 15 spaces for blanking out purposes.

Lines 300–370: These lines specify whose turn it is, how many numbers will be flashed, and print the ready prompt.

Lines 380–570: This series asks the appropriate player to repeat the sequence of numbers and checks it to see if it is correct or not. It then increases the level of difficulty if the response was correct.

Lines 1000–1030: This routine produces the flashing "GET READY" prompt.

Lines 2000–2050: This subroutine displays the congratulatory message "YOU GOT IT" if the response was correct.

Lines 3000–3100: This subroutine displays the message telling the player of a wrong response; then the computer flashes the correct sequence.

Lines 4000–4110: This segment checks to see if there are any players left in the game. If not, it displays all the scores.

Possible Adaptations

1. The input could be timed by inserting the appropriate instructions between lines 470 and 480.
2. If the pace of the game is too fast it may be slowed down by changing the delay time implemented in the **FOR/NEXT** loop in lines 430–440.

Chapter 20

LOTTERY

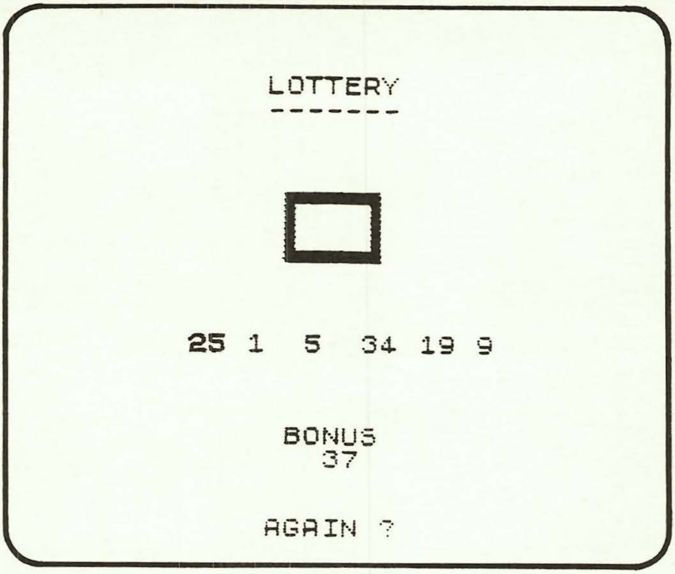
- The Game
- Program 20: LOTTERY
- Points of Interest
- Possible Adaptations

The Game

Many of the state lotteries that are so popular nowadays require that the bettor select six numbers. These numbers are matched against six numbers "drawn from a hat" (so to speak), and those who successfully chose those numbers are the winners. In New York State the maximum number allowed is 44, while in New Jersey it is 36. In other words, in New York the selected numbers may range from 1 to 44 and in New Jersey from 1 to 36. Some of the lotteries allow for a so-called bonus number. Since the chances of winning a lottery are so small and in light of its popularity, the prizes usually awarded often run into the millions of dollars.

In this T/S version of the lottery the player decides what the upper limit is. This can range from 10 to 99. Any number outside this range is automatically ignored and another selection is requested. The player then decides how many such numbers the computer should randomly select. The range this time is from 1 to 20. Next, the player is asked if he or she wants a bonus number to be selected by the computer; if so, one is made.

Before the game is commenced the player makes his or her predictions on a piece of paper. Then, when the program



is run, the computer's selections are matched against the player's. Some kind of point system can easily be devised. Within a rectangular box a series of generated numbers is shown. It is the last one shown that is a final selection. Once the required number of selections is made, they are displayed in a horizontal row beneath the box (together with the bonus number, if one was requested) and the player is asked if another round is required. The display, incidentally, is always centered on the screen; this is done by some fancy footwork in the program.

So you see, by using this program, you can get plenty of practice in selecting winning lottery numbers. Remember, if you use this program to select the winning entries to any real lottery, you are morally bound to buy all of your relatives and friends a brand new T/S computer!

Program 20: LOTTERY

```
1ØØ  DIM N(99)
11Ø  GOSUB 1ØØØ
```

```
120  PRINT AT 10,4;"ENTER_UPPER_LIMIT_10-99"
130  INPUT MAX
135  IF MAX>99 OR MAX<10 THEN GOTO 120
140  PRINT AT 10,4;"HOW_MANY_NUMBERS?,
     _1-20"
150  INPUT Q
155  IF Q>MAX OR Q<1 OR Q>20 THEN GOTO 140
160  PRINT AT 10,1;"DO_YOU_WANT_ANY_BONUS
     _NUMBER?"
170  PRINT AT 11,9;"TYPE_Y_OR_N"
180  INPUT B$
200  FOR I=1 TO MAX
210  LET N(I)=0
220  NEXT I
230  GOSUB 1000
240  GOSUB 2000
290  FOR I=1 TO Q
300  FOR K=1 TO 5
310  LET Z=INT (RND*MAX)+1
320  IF N(Z)=1 THEN GOTO 310
330  IF K=5 THEN LET N(Z)=1
340  PRINT AT 7,14;Z;"_"
350  NEXT K
355  PRINT AT 7,14;"_ _"
360  GOSUB 3000
370  NEXT I
380  IF B$="N" THEN GOTO 450
390  LET Z=INT (RND*MAX)+1
400  IF N(Z)=1 THEN GOTO 390
410  PRINT AT 16,13;"BONUS"; AT 17,15;Z
```

```
 45Ø  PRINT AT 2Ø,12;"AGAIN_?"
 46Ø  INPUT A$
 47Ø  IF A$="Y" THEN GOTO 2ØØ
 48Ø  STOP
1ØØØ  CLS
1Ø1Ø  PRINT AT 1,12;"LOTTERY"; AT 2,12;"--------"
1Ø2Ø  RETURN
2ØØØ  PRINT AT 6,13;"▛▀▀▀▜"
2Ø1Ø  PRINT AT 7,13;"▌_ _ _▐"
2Ø2Ø  PRINT AT 8,13;"▙▄▄▄▟"
2Ø3Ø  LET H=Q
2Ø4Ø  IF H<1Ø THEN GOTO 21ØØ
2Ø5Ø  LET H=H-1Ø
2Ø6Ø  FOR I=1 TO 28 STEP 3
2Ø7Ø  PRINT AT 14,I;"██"
2Ø8Ø  NEXT I
2Ø9Ø  IF H=Ø THEN RETURN
21ØØ  LET X=INT ((31-(H*2+(H-1)))/2)
211Ø  FOR I=1 TO (H*2+(H-2)) STEP 3
212Ø  PRINT AT 12,X+I;"██"
213Ø  NEXT I
214Ø  RETURN
3ØØØ  IF I>1Ø THEN GOTO 31ØØ
3Ø1Ø  PRINT AT 12,X+((I-1)*3+1);Z;"_"
3Ø2Ø  RETURN
31ØØ  PRINT AT 14,(I-11)*3+1;Z;"_"
311Ø  RETURN
```

Points of Interest

Lines 120–180: These lines not only prompt and accept the user-specified boundaries, but also validate all numeric data.

Lines 200–220: These lines set the array N to zero to prepare for the avoidance of the possibility that any randomly selected number is repeated. This is equivalent to removing the number from the pool once it has been selected.

Lines 230–240: These two subroutines set up the initial display.

Lines 290–370: These lines produce the "shuffling" of the randomly selected numbers (those that appear in quick succession in the rectangular box) and select the last one generated.

Lines 380–410: If the option to select a bonus number is elected, these lines produce and display it.

Lines 450–480: Provision is made here for repeated use of the program.

Lines 1000–1020: This is a subroutine to produce the heading.

Lines 2000–2140: These lines center the "pigeon holes" (the rectangular box) for the display of the numbers.

Lines 3000–3110: This series displays the actual numbers neatly centered.

Possible Adaptations

1. The boundary limits can be changed to allow for wider use; should the desired maximum be increased beyond 99, the coding to display the selected number will have to be modified.
2. It might be desired to have all of the "shuffling" shown on the display at one time. This would make for a radical change to the program.

Chapter 21

GAMBLER

- The Game
- Program 21: GAMBLER
- Points of Interest
- Possible Adaptations

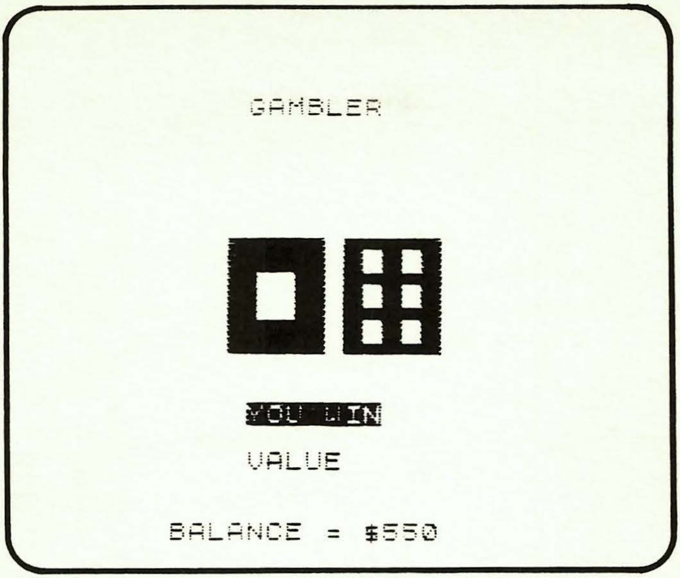
The Game

This is a fast-paced graphics game similar in style to the familiar game of "craps." The rules for the game are:

- 7 or 11 on the first roll is a win.
- 2 or 3 on the first roll is a loss.
- Any number on the first roll becomes your "mark." You must then roll the dice continually until the mark is rolled again. If a 7 is rolled before the mark, however, you lose.

In this version of the game a full betting routine is implemented and the player starts out with a balance of $500. As the game proceeds, winning and losing bets change the value of the balance and this value is displayed at the bottom of the screen. The value of the bet is always compared against the current balance. If the bet exceeds the balance, it is ignored and a request is made for another bet. A bet of zero terminates the game, flashing the message GAME OVER.

The rolling of the dice is simulated through the ordinary graphics facility provided on your T/S computer. Rather than clearing the whole screen between throws of the dice, only selected portions of it are blanked out; in particular, only the dice themselves.



Program 21: GAMBLER

```
100  LET BAL=500
110  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
120  LET R=8
130  LET M=0
140  CLS
150  PRINT AT 3,12;"GAMBLER"
170  GOSUB 2000
180  LET C=11
185  LET T=0
190  FOR Z=1 TO 2
200  LET X=INT (RND*6)+1
210  IF Z=1 THEN GOTO 230
220  LET C=17
230  FOR Q=1 TO X
```

```
24Ø  GOSUB (Q+2)*1ØØ+7ØØ
25Ø  NEXT Q
26Ø  LET T=T+X
27Ø  NEXT Z
28Ø  LET M=M+1
29Ø  PRINT AT 18,12;"VALUE_";T;"_"
3ØØ  PAUSE 1ØØ
3Ø5  PRINT AT 18,18;"_ _"
31Ø  IF M<>1 THEN GOTO 4ØØ
32Ø  IF T=7 OR T=11 THEN GOTO 5ØØ
33Ø  IF T=2 OR T=3 THEN GOTO 6ØØ
34Ø  IF M<>1 THEN GOTO 4ØØ
35Ø  LET HOLD=T
36Ø  PRINT AT 2Ø,12;"MARK_";HOLD
37Ø  GOTO 18Ø
4ØØ  IF HOLD=T THEN GOTO 5ØØ
41Ø  IF T=7 THEN GOTO 6ØØ
43Ø  GOTO 18Ø
5ØØ  FOR I=1 TO 1Ø
51Ø  PRINT AT 16,12;"YOU_WIN";AT 16,12;"YOU
     WIN"
52Ø  NEXT I
53Ø  LET P=1
54Ø  GOTO 7ØØ
6ØØ  FOR I=1 TO 1Ø
61Ø  PRINT AT 16,12;"YOU_LOSE";AT 16,12;"YOU
     LOSE"
62Ø  NEXT I
63Ø  LET P=-1
```

```
700  LET BAL=BAL+P*BET
710  PRINT AT 21,8;"BALANCE_=_$";BAL;"_ _
     _ _ _"
720  PAUSE 100
725  IF BAL<=0 THEN GOTO 3000
730  PRINT AT 16,12;S$
735  PRINT AT 18,12;S$
740  FOR I=9 TO 13
750  PRINT AT I,11;S$
760  NEXT I
770  LET M=0
780  GOTO 170
1000 PRINT AT R+1,C;"█████"
1010 PRINT AT R+2,C;"█▛▀▜█"
1020 PRINT AT R+3,C;"█▌_▐█"
1030 PRINT AT R+4,C;"█▙▄▟█"
1040 PRINT AT R+5,C;"█████"
1050 RETURN
1100 PRINT AT R+1,C;"▛▀███"
1110 PRINT AT R+2,C;"▌_███"
1120 PRINT AT R+3,C;"█████"
1130 PRINT AT R+4,C;"███_▐"
1140 PRINT AT R+5,C;"███▄▟"
1150 RETURN
1200 PRINT AT R+1,C;"▛▛███"
1210 PRINT AT R+2,C;"▟▟███"
1220 PRINT AT R+3,C;"██_██"
1230 PRINT AT R+4,C;"███▛▛"
1240 PRINT AT R+5,C;"███▟▟"
```

```
125Ø  RETURN
13ØØ  PRINT AT R+1,C;"█████"
131Ø  PRINT AT R+2,C;"█_█_█"
132Ø  PRINT AT R+3,C;"█████"
133Ø  PRINT AT R+4,C;"█_█_█"
134Ø  PRINT AT R+5,C;"█████"
135Ø  RETURN
14ØØ  PRINT AT R+1,C;"▛▜█▛▜"
141Ø  PRINT AT R+2,C;"▙▟█▙▟"
142Ø  PRINT AT R+3,C;"██_██"
143Ø  PRINT AT R+4,C;"▛▜█▛▜"
144Ø  PRINT AT R+5,C;"▙▟█▙▟"
145Ø  RETURN
15ØØ  PRINT AT R+1,C;"█▀█▀█"
151Ø  PRINT AT R+2,C;"█▄█▄█"
152Ø  PRINT AT R+3,C;"█_█_█"
153Ø  PRINT AT R+4,C;"█▀█▀█"
154Ø  PRINT AT R+5,C;"█▄█▄█"
155Ø  RETURN
2ØØØ  PRINT AT 2Ø,8;"PLACE_YOUR_BET"
2Ø1Ø  INPUT BET
2Ø2Ø  PRINT AT 2Ø,8;S$
2Ø3Ø  IF BET<=BAL AND BET>=1 THEN RETURN
2Ø4Ø  IF BET=Ø THEN GOTO 3ØØØ
2Ø5Ø  GOTO 2ØØØ
3ØØØ  FOR I=1 TO 1Ø
3Ø1Ø  PRINT AT 2Ø,11;"GAME_OVER";AT 2Ø,11;
      "GAME OVER"
3Ø2Ø  NEXT I
```

Points of Interest

Line 100: This line starts the game out with a balance of $500.

Line 110: This line uses a sequence of 15 blanks for use in several different points in the program.

Lines 140–300: These lines display the whole game board.

Lines 730–760: These lines blank out the dice in preparation for the next throw.

Lines 1000–1550: This segment handles the display of the six faces of the die.

Lines 2000–2050: These lines implement the full betting routine.

Lines 3000–3020: This series displays the flashing "GAME OVER" message.

Possible Adaptations

1. The starting balance can always be changed by amending line 100.
2. If the patience and artistic talent are there, the dice designs can always be refined.

Chapter 22

BRAIN STUMPER

- The Game
- Program 22: BRAIN STUMPER
- Points of Interest
- Possible Adaptations

The Game

In this game we shall test your skills in elementary arithmetic. However, there really isn't any need to get terribly nervous because all the program does is to exercise your ability to add, subtract, multiply, and divide. The process selected is not up to you, however, for the computer decides which one to select. Once the selection is made, you are asked to identify one of four possible answers. All you do is type in 1, 2, 3 or 4, depending on what you think the correct answer is. With a little care you should be able to answer all the questions right. In the event that you give a wrong response, the computer obligingly displays the correct answer. On the other hand, should your answer be correct, you are awarded a point. In order to keep as many of the questions on the screen as possible, a scrolling technique is used. At the end of the game, your score based on a percentage is displayed.

What is so convenient about this program is that each time the program is run, the values selected for calculation will be different because they are randomly selected by the computer. So there is no danger of learning the correct answers by heart. The choice of the correct answer is made in a multiple-choice format, but the alternatives have been arranged so that each

```
            BRAIN STUMPER

1.     DIVIDE     154 / 7

1. 12  2. 22  3. 17  4. 27

2.     MULTIPLY 13 * 19

1. 247  2. 242  3. 232  4. 252

3.     DIVIDE     198 / 11

1. 3  2. 8  3. 18  4. 13

4.     DIVIDE     165 / 11

1. 0  2. 10  3. 15  4. 20

        CHOOSE 1,2,3 OR 4
         SCORE: 3 / 3
```

one of them could easily be mistaken for the correct one. As a result, you really have to be on your toes throughout the quiz. Since there is no time limit, you may take your time in answering the questions.

Program 22: BRAIN STUMPER

```
100  LET PT=Ø
110  LET TL=Ø
120  DIM X(4)
130  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
140  GOSUB 1ØØØ
200  LET Q=INT (RND*4+1)
210  LET TL=TL+1
220  GOSUB Q*1ØØ+1ØØØ
230  LET ANS=INT (RND*4)+1
240  LET X(ANS)=A
```

```
 25Ø  GOSUB 15ØØ
 26Ø  PRINT AT 16,1;TL;".";TAB (6);Z$;TAB
      (15);N1;Y$;N2
 3ØØ  PRINT AT 18,1;
 31Ø  FOR K=1 TO 4
 32Ø  PRINT K;"._";X(K);"_ _";
 33Ø  NEXT K
 34Ø  PRINT AT 2Ø,7;"CHOOSE_1,2,3_OR_4"
 35Ø  INPUT CHOICE
 36Ø  PRINT AT 2Ø,7;S$;"_ _"
 365  IF CHOICE<Ø OR CHOICE>4 THEN GOTO 34Ø
 37Ø  IF CHOICE=Ø THEN GOTO 2ØØØ
 38Ø  IF CHOICE=ANS THEN GOTO 6ØØ
 4ØØ  PRINT AT 19,11;"INCORRECT";AT 2Ø,4;
      "ANSWER_IS_";X(ANS);"_CHOICE_";ANS
 41Ø  PAUSE 75
 42Ø  PRINT AT 19,11;S$;AT 2Ø,1;S$;S$
 44Ø  GOTO 14Ø
 6ØØ  LET PT=PT+1
 61Ø  FOR I=1 TO 5
 62Ø  PRINT AT 19,12;"CORRECT";AT 19,12;
      "CORRECT";AT 19,12;S$
 63Ø  NEXT I
 64Ø  GOTO 14Ø
1ØØØ  PRINT AT 2Ø,1;S$;S$
1Ø1Ø  PRINT AT 21,1;S$;S$
1Ø15  FOR G=1 TO 4
1Ø2Ø  SCROLL
1Ø3Ø  NEXT G
1Ø35  PRINT AT Ø,1;S$;S$
```

```
1Ø4Ø  PRINT AT 1,1;S$;S$
1Ø5Ø  PRINT AT 2,1;S$;S$
1Ø6Ø  PRINT AT 1,9;"BRAIN_STUMPER"
1Ø7Ø  PRINT AT 21,8;"SCORE:_";PT;"_/_";TL
1Ø8Ø  RETURN
11ØØ  LET Z$="ADD"
111Ø  LET N1=INT (RND*9Ø)+1Ø
112Ø  LET N2=INT (RND*9Ø)+1Ø
113Ø  LET A=N1+N2
114Ø  LET Y$="_+_"
115Ø  RETURN
12ØØ  LET Z$="SUBTRACT"
121Ø  LET N1=INT (RND*9Ø)+1ØØ
122Ø  LET N2=INT (RND*9Ø)+1Ø
123Ø  LET A=N1-N2
124Ø  LET Y$="_-_"
125Ø  RETURN
13ØØ  LET Z$="MULTIPLY"
131Ø  LET N1=INT (RND*2Ø)+5
132Ø  LET N2=INT (RND*2Ø)+5
133Ø  LET A=N1*N2
134Ø  LET Y$="_*_"
135Ø  RETURN
14ØØ  LET Z$="DIVIDE"
141Ø  LET N2=INT (RND*1Ø)+3
142Ø  LET N1=INT (RND*2Ø+1Ø)*N2
143Ø  LET A=N1/N2
144Ø  LET Y$="_/_"
145Ø  RETURN
15ØØ  FOR K=1 TO 4
```

```
1510  IF X(K)=A THEN GOTO 1570
1520  LET X(K)=INT (RND*(K+1)-3)*5+A
1530  FOR M=1 TO K-1
1540  IF X(M)=X(K) THEN GOTO 1520
1550  NEXT M
1560  IF X(K)=A THEN GOTO 1520
1570  NEXT K
1580  RETURN
2000  LET SCR1=PT/(TL-1)*100
2005  LET SCR2=INT (SCR1)
2010  FOR I=1 TO 10
2020  PRINT AT 19,10;S$;AT 20,14;"_ _ _"
2030  PRINT AT 19,10;"FINAL_SCORE";AT 20,14;SCR2
2040  NEXT I
```

Points of Interest

Line 130: This is a line of 15 blanks to assist in the scrolling effect.

Line 200: This line generates a random number between 1 and 4 that directs the program to the appropriate mathematical operation.

Line 220: In this line, a calculated **GOSUB** accesses the appropriate subroutine for the particular arithmetic operation.

Lines 230–240: These lines select one of the four choices as the answer.

Lines 260–640: These lines print (display) the question, pose the possible choices and accept the player's response.

Lines 1000–1080: This subroutine handles the entire scrolling function by blanking out and replacing lines.

Lines 1100–1150: This subroutine gives the process for addition.

Lines 1200–1250: This subroutine handles the subtraction process.

Lines 1300–1350: This subroutine directs the multiplication process.

Lines 1400–1450: This subroutine handles the division process.

Lines 1500–1580: This segment creates possible answers to resemble the correct answer.

Lines 2000–2040: This group takes the player's final score and converts it to percentage form and displays it.

Possible Adaptations

1. It would make for an even more challenging game if the responses not only had to be correct, but had to be entered within a limited time period. This could be implemented using the, by now, familiar **PAUSE** and **INKEY$** statements.
2. The points scored for each correct response could, of course, be changed at will. Perhaps a penalty could be applied for an incorrect response!

Chapter 23

MAGIC SQUARES

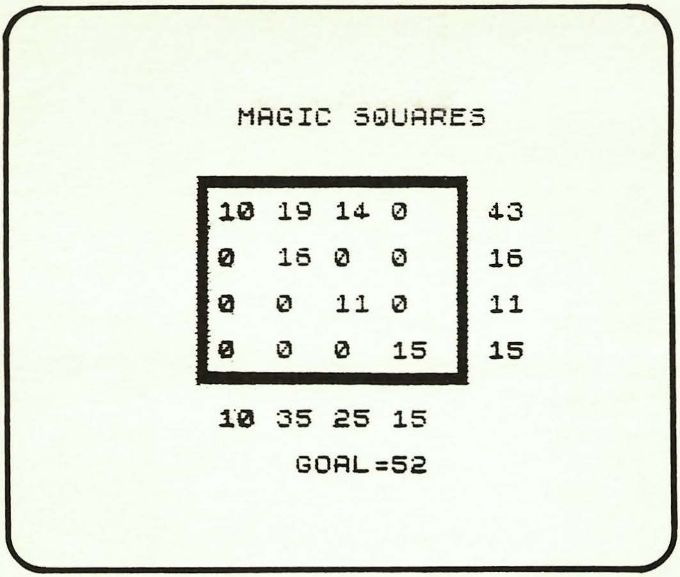
- The Game
- Program 23: MAGIC SQUARES
- Points of Interest
- Possible Adaptations

The Game

This game is geared toward the mathematically oriented. It is only fair to say that it is rather tricky to play and can be extremely time-consuming. The idea behind the game is to fill in numbers in a 4 × 4 matrix board such that all the rows and columns add up to a computer-selected arbitrary number.

You are allowed to specify the location in which to place a number in the normal coordinate manner. (Any improper coordinates are ignored and another opportunity is given.) You then specify the particular number you wish to be placed in that location. Although a value of 0 is unacceptable, it may be used as a temporary measure to hold a particular location while you are engaged in switching other numbers around. So long as any zeros remain on the grid, the problem is not solved. Should you wish to change your mind after specifying a particular number, you are always at liberty to enter another number in any given location.

The program supplies automatic summation of the rows and columns as soon as a number is entered, leaving your mind free to concentrate on the more complicated aspects of the game. Should your selection of a number be greater than the arbitrarily selected goal set by the computer—called the SUM in the program—that number is automatically rejected



because it is an obvious error. For that matter, so is any negative number rejected. In either case, another value is requested. The game starts out with four numbers located on the main diagonal; this is the diagonal that extends from the top left to the lower right. The sum of these numbers is the "goal" of the game. You are not allowed to change them or to use any number more than once.

Program 23: MAGIC SQUARES

```
100 DIM X(4,4)
105 DIM R(4)
110 DIM C(4)
115 GOSUB 1000
120 LET SUM=0
130 FOR I=1 TO 4
140 LET SUM=SUM+X(I,I)
150 NEXT I
```

```
 160  PRINT AT 17,12;"GOAL=";SUM
 170  GOSUB 2000
 180  PRINT AT 19,11;"CO-ORDS_?"
 190  INPUT N$
 200  PRINT AT 19,11;"________"
 210  IF N$(1)=N$(2) OR N$>"44" OR N$<"11"
      THEN GOTO 180
 220  LET Y=VAL N$(1)
 230  LET Z=VAL N$(2)
 240  PRINT AT 19,11;"VALUE_?"
 250  INPUT V1
 260  PRINT AT 19,11;"_______"
 270  IF V1>SUM OR V1<0 THEN GOTO 240
 300  FOR I=1 TO 4
 310  FOR J=1 TO 4
 320  IF V1=X(I,J) THEN GOTO 240
 330  NEXT J
 340  NEXT I
 350  LET X(Y,Z)=V1
 360  GOSUB 2000
 370  GOSUB 3000
 380  GOTO 180
1000  CLS
1010  PRINT AT 2,9;"MAGIC_SQUARES"
1020  PRINT AT 5,7;"▙▀▀▀▀▀▀▀▀▀▀▀
      ▀▜"
1030  FOR I=6 TO 12
1040  PRINT AT I,7;"▌"; TAB (20);"▐"
1050  NEXT I
```

```
1Ø6Ø  PRINT AT 13,7;"▙▄▄▄▄▄▄▄▄▄▄
      ▄▄▟"
11ØØ  FOR I=1 TO 4
111Ø  FOR J=1 TO 4
112Ø  IF I< >J THEN GOTO 12ØØ
113Ø  LET Y=INT (RND*9)+1Ø
114Ø  FOR K=1 TO I-1
115Ø  IF X(K,K)=Y THEN GOTO 113Ø
116Ø  NEXT K
117Ø  LET X(I,J)=Y
118Ø  GOTO 121Ø
12ØØ  LET X(I,J)=Ø
121Ø  NEXT J
122Ø  NEXT I
123Ø  RETURN
2ØØØ  FOR I=1 TO 4
2Ø1Ø  LET R(I)=Ø
2Ø2Ø  LET C(I)=Ø
2Ø3Ø  FOR J=1 TO 4
2Ø4Ø  LET R(I)=R(I)+X(I,J)
2Ø5Ø  LET C(I)=C(I)+X(J,I)
2Ø7Ø  PRINT AT (I-1)*2+6,(J-1)*3+8;X(I,J);"_"
2Ø8Ø  NEXT J
2Ø9Ø  NEXT I
21ØØ  LET S=1
211Ø  FOR K=6 TO 12 STEP 2
212Ø  PRINT AT K,22;R(S);"_"
213Ø  PRINT AT 15,K+S+1;C(S);"_"
214Ø  LET S=S+1
```

```
2150  NEXT K
2160  RETURN
3000  FOR I=1 TO 4
3010  IF R(I)< > SUM OR C(I)< >SUM THEN RETURN
3020  FOR J= TO 4
3030  IF X(I,J)=0 THEN RETURN
3040  NEXT J
3050  NEXT I
3100  FOR I=1 TO 10
3110  PRINT AT 19,11;"YOU_GOT_IT";AT 19,11;
      "YOU GOT IT"
3120  NEXT I
```

Points of Interest

Lines 100–110: These lines reserve space for the matrix and the two arrays.

Line 115: This line directs the program to the subroutine where the preliminary display is produced and the four numbers constituting the main diagonal are produced.

Lines 120–160: This group of lines sums and displays the goal.

Lines 180–270: These lines prompt the player for the coordinates which are entered as a double digit in string format. They are immediately validated.

Lines 300–350: These lines place the value selected in the proper place in the 4 × 4 grid.

Lines 1000–1230: This subroutine prints the header and randomly selects the numbers to lie on the main diagonal. The program ensures that no number is repeated.

Lines 2000–2160: This subroutine tabulates the row and column totals.

Lines 3000–3120: These lines determine if the square is complete; if so, the message "YOU GOT IT" is flashed. If not, the game continues in the normal way.

Possible Adaptations

1. A rather difficult amendment would be to arrange for the option that the computer supplies the next move in the event the player is stuck.
2. The size of the grid could be changed, depending upon the desired level of difficulty.
3. To add to the difficulty, a time limit could be imposed.

Chapter 24

JET FIGHTER

- The Game
- Program 24: JET FIGHTER
- Points of Interest
- Possible Adaptations

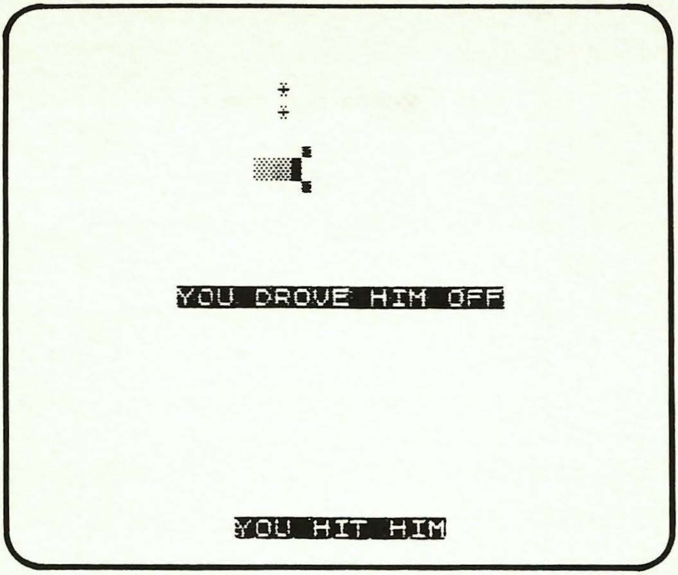
The Game

You are now a jet fighter in the midst of the fiercest, wildest, most challenging air duel of your entire career. An enemy jet is flying directly above you. Although the speeds change during the course of the game, your speed is always exactly half of that of your enemy. You will recognize the enemy jet since it is shown in black on the screen, while yours is shaded gray. Every time the jets cross paths, the enemy will try to riddle your jet full of holes. If you are quick you might be able to blast him to kingdom come before you are fired upon.

To fire a shot, use the F key on the keyboard. If you are successful in hitting the enemy jet it will be damaged, and will retreat upwards on the screen with you in hot pursuit. However, if the enemy hits your jet it will have the effect of causing your jet to fly at a lower level while he chases you. The game has two possible outcomes: either you succeed in driving the enemy away—in which case you become the victorious master of the skies—or he manages to irreparably damage your jet and force you to land, in which case, of course, you lose the game. So go to it!

Program 24: JET FIGHTER

```
100  LET C1=8
110  LET C2=3
```




```
12Ø  LET Y1=11
13Ø  LET Y2=29
14Ø  LET S$="_ _ _"
15Ø  CLS
16Ø  GOSUB 1ØØØ
17Ø  LET X=VAL "24" (INT (RND*1)+1)
175  LET C2=C2+X
18Ø  IF C2>29 THEN LET C2=3
19Ø  LET Y2=Y2-X/2
2ØØ  IF Y2<2 THEN LET Y2=29
21Ø  GOSUB 2ØØØ
22Ø  IF INKEY$<>"" THEN GOSUB 3ØØØ
23Ø  IF C2<>Y2 THEN GOTO 16Ø
24Ø  PAUSE 1Ø
25Ø  IF INKEY$="F" THEN GOTO 5ØØ
```

```
260   GOSUB 4000
270   PAUSE 10
280   GOSUB 5000
290   GOTO 150
500   PRINT AT C1,C2+1;"▒";AT C1,C2+1;"*";
      AT C1+1,C2+1;"▒";AT C1+1,C2+1;"*"
510   FOR I=1 TO 7
520   PRINT AT 21,10;"YOU_HIT_HIM";AT 21,10;
      "YOU HIT HIM"
530   NEXT I
540   LET C1=C1-2
550   LET Y1=Y1-2
560   GOSUB 5000
570   GOTO 150
1000  PRINT AT C1,C2;S$
1010  PRINT AT C1+1,C2;S$
1020  PRINT AT Y1,Y2;S$
1030  PRINT AT Y1+1,Y2;S$
1040  RETURN
2000  PRINT AT C1,C2;"▚▄▄"
2010  PRINT AT C1+1,C2;"▙▀▀"
2020  PRINT AT Y1,Y2;"▄▄▞"
2030  PRINT AT Y1+1,Y2;"▀▀▚"
2040  RETURN
3000  PRINT AT Y1-1,Y2+1;"▒";AT
      Y1-1,Y2+1;"*";AT Y1-1,Y2+1;"_"
3010  RETURN
4000  PRINT AT Y1,Y2+1;"▒";AT Y1,Y2+1;"+";
      AT Y1+1,Y2+1;"▒";AT Y1+1,Y2+1;"+"
4010  FOR I=1 TO 7
```

```
4020  PRINT AT 21,10;"YOU_WERE_HIT";AT 21,10;
      "YOU WERE HIT"
4030  NEXT I
4040  LET C1=C1+2
4050  LET Y1=Y1+2
5000  IF C1<1 THEN GOTO 5100
5010  IF Y1>19 THEN GOTO 5200
5030  RETURN
5100  PRINT AT C1+2,C2;"_*_";AT
      C1+2,C2;"_*_"
5110  PRINT AT C1+3,C2;"_*_";AT
      C1+3,C2;"_*_"
5120  FOR I=1 TO 10
5130  PRINT AT 11,7;"YOU_DROVE_HIM_OFF";
      AT 11,7;"YOU DROVE HIM OFF"
5140  NEXT I
5150  STOP
5200  PRINT AT Y1-2,Y2;"_+_";AT Y1-2,Y2;"_+_"
5210  PRINT AT Y1-1,Y2;"_+_";AT Y1-1,Y2;"_+_"
5220  FOR I=1 TO 10
5230  PRINT AT
      11,6;"YOU_WERE_SHOT_DOWN";AT 11,6;
      "YOU WERE SHOT DOWN"
5240  NEXT I
5250  STOP
```

Points of Interest

Line 170: This line randomly extracts a string of numbers to determine the speed.

Lines 175–200: These lines set up the player's speed to be one half of the computer's.

Lines 220–290, 500–570: These segments process the F key when pressed.

Lines 1000–1040: This subroutine blanks out the previous position.

Lines 2000–2040: This subroutine prints the current position of both jets.

Lines 3000–3010: This is a subroutine to handle the displaying of a shot that misses.

Lines 4000–5030: This subroutine handles the situation when the enemy fires and hits the player's jet. This section of code also determines if the game is over.

Lines 5100–5150: This set outputs the player's winning display.

Lines 5200–5250: These lines output the computer's winning display.

Possible Adaptations

1. The relative speeds can be changed.

Chapter 25

INTRUDERS

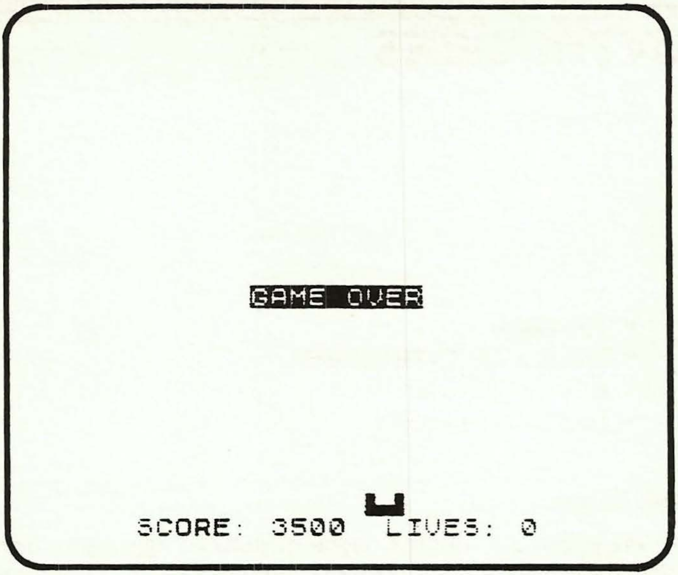
- The Game
- Program 25: INTRUDERS
- Points of Interest
- Possible Adaptations

The Game

In this game, six different types of galactic aggressors have banded together and are planning to invade Earth. In order to pursue their aim, they dispatch small scout ships which gather information and relay it back to their mother ships. It is at that point that the scout ships glide to Earth in the form of guided missiles. Your mission as the exalted guardian of the Earth is to catch these falling ships and prevent them from crashing into major population centers. For this purpose you are provided with a super-catcher that can move to the left and to the right: use the 5-key for movement to the left, and the 8-key for the movement to the right. For rapid movement use the H-key, which will jettison you toward hyperspace.

If your aim is successful, you will observe the attacking ship trapped by your catcher. If you miss, you are debited one of the five ''lives'' you have.

It is only fair to warn you that the falling ships are not only of variable size, but also do not always fall in a straight line. They could fall straight down in a vertical direction, diagonally to the left, or diagonally to the right. This is under the sole control of the computer. Points are awarded according to the smallness of the objects hit. They vary from 100 points for the largest to 600 for the smallest intruders. So go to it; remember the free world is depending on you!



Program 25: INTRUDERS

```
100  LET S$="_ _ _ _ _ _ _ _ _ _ _"
105  LET Q$="ØØ.5.5"
110  LET X$="▞▒▄█▀▝"
120  LET PT=Ø
130  LET MISS=Ø
140  LET C=15
150  LET D$="▙▟"
160  PRINT AT 2Ø,C;D$
170  LET Y=INT (RND*28)+2
180  LET Z=INT (RND*6)+1
190  IF Y<12 THEN GOTO 3ØØ
200  IF Y>2Ø THEN GOTO 4ØØ
300  GOSUB 1ØØØ
310  GOSUB 2ØØØ
```

```
320  GOTO 170
400  GOSUB 1000
410  LET N=-N
420  GOSUB 2000
430  GOTO 170
1000 LET Q=INT (RND*3)+1
1010 LET N=VAL Q$((Q-1)*2+1 TO (Q-1)*2+2)
1020 RETURN
2000 FOR I=1 TO 20
2010 PRINT AT I-1,Y;S$(1)
2020 LET Y=Y+N
2030 PRINT AT I,Y;X$(Z)
2040 IF INKEY$<>"" THEN GOSUB 3000
2050 NEXT I
2060 PRINT AT I-1,Y;S$(1)
2065 PRINT AT 20,C;D$
2070 IF C=Y OR C+1=Y THEN LET PT=PT+Z*100
2080 IF C<>Y AND C+1<>Y THEN GOSUB 4000
2090 PRINT AT 21,5;"SCORE:_";PT;
     "_ _LIVES:_";5-MISS
2100 RETURN
3000 LET W$=INKEY$
3010 IF W$="5" THEN GOTO 3100
3020 IF W$="8" THEN GOTO 3200
3030 IF W$="H" THEN GOTO 3300
3040 RETURN
3100 LET M1=-1
3110 GOTO 3400
3200 LET M1=1
3210 GOTO 3400
```

```
3300  PRINT AT 20,C;S$(1 TO 2)
3310  LET C=INT (RND*30)+1
3320  GOTO 3430
3400  IF C+M1>30 OR C+M1<1 THEN RETURN
3410  PRINT AT 20,C;S$(1 TO 2)
3420  LET C=C+M1
3430  PRINT AT 20,C;D$
3440  RETURN
4000  LET MISS=MISS+1
4010  FOR I=1 TO 7
4020  PRINT AT 11,11;"YOU_MISSED";AT 11,11;
      "YOU MISSED";AT 11,11;S$
4030  NEXT I
4040  IF MISS<5 THEN RETURN
4050  FOR I=1 TO 10
4060  PRINT AT 11,11;"GAME_OVER";AT 11,11;
      "GAME OVER"
4070  NEXT I
4080  PRINT AT 21,5;"SCORE:_";PT;
      "_ _LIVES:_";5-MISS
```

Points of Interest

Line 100: S$ is a row of 10 spaces for blanking out purposes.

Line 105: Q$ is a string of values used for the angle of descent.

Line 110: X$ is the string containing the six various intruder shapes.

Lines 150–430: This is the control routine for the descent of the intruder and the screen movement of the catcher.

Lines 1000–1020: The descent angle is randomly selected by this code.

Lines 2000–2100: These lines process the movement of the intruder as well as determine if the intruder is successfully caught; they also display the score and the remaining lives.

Lines 3000–3440: This series handles the effects of pressing the 5, 8, and H keys.

Lines 4000–4080: These lines display the flashing messages and determine if the game is over.

Possible Adaptations

1. The number of lives can easily be changed.
2. The shapes of the intruders may be changed.
3. You might wish to add to the six different intruder shapes.

PART III

Games for the Timex/Sinclair 2068

The seven games in this part can be used only on the T/S 2068. These programs were specially designed to use the T/S 2068 to its fullest potential. They employ user-defined graphics, color, and sound, and two of the programs even have a joystick option. All of these games should be programmed in the uppercase mode; this is accessed by the caps-lock key or (C)shifted-2. The keyboard and commands are quite different from those of the previous models; refer to the Appendixes at the end of this section.

Chapter 26

COLOR ORGAN

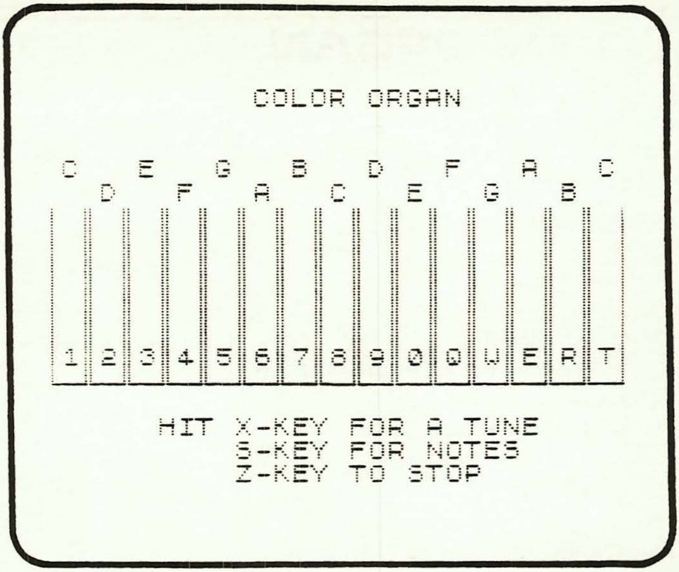
- The Game
- Program 26: COLOR ORGAN
- Points of Interest
- Possible Adaptations

The Game

This program simulates all the white keys of a two-octave organ. The lowest key is middle C. When the program is run, the screen is set up and the program title is displayed for a few seconds; then the keyboard appears. Across the top, you'll see letters representing the notes C, D, E, and so on. Across the bottom are the keys that play these notes. To play any note, just hit the appropriate key. Pressing the X key plays a built-in tune which is sure to be familiar to you. Typing in S plays the scale of available notes, while pressing Z stops the program.

Program 26: COLOR ORGAN

```
100  CLS
110  PRINT AT 8,11;"COLOR_ORGAN"
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
130  PRINT AT 13,13;"(c)-1984"
140  FOR I=1 TO 2
150  FOR J=0 TO 6
```




```
160  BORDER J: PAUSE 2Ø
17Ø  NEXT J
18Ø  NEXT I
19Ø  DIM N(15)
195  DIM C(15)
2ØØ  DIM L$(15)
21Ø  CLS
22Ø  FOR I=12 TO 246 STEP 16
23Ø  PLOT I,118
24Ø  DRAW Ø,-6Ø
25Ø  DRAW 14,Ø
26Ø  DRAW Ø,6Ø
27Ø  NEXT I
275  LET Q$="C_E_G_B_D_F_A_C"
```

```
276  LET W$="_D_F_A_C_E_G_B_"
28Ø  LET X$="123456789ØQWERT"
29Ø  FOR I=1 TO 15
295  PRINT AT 5,2*I;Q$(I);AT 6,2*I;W$(I);AT 13,2*I;X$(I)
3ØØ  READ N(I),C(I)
3Ø5  NEXT I
3Ø6  FLASH 1
3Ø7  PAPER 6
31Ø  PRINT AT 2,12;"COLOR_ORGAN"
311  FLASH Ø
312  PAPER 7
315  PRINT AT 16,7;"HIT_X-KEY_FOR_A_TUNE";
     AT 17,11;"S-KEY_FOR_NOTES";AT 18,11;"Z-
     KEY_TO_STOP"
32Ø  DATA Ø,Ø,2,1,4,2,5,3,7,4,9,5,11,6,12,Ø,14,1,16,2,17,
     3,19,4,21,5,23,6,24,7
33Ø  IF INKEY$=" " THEN GO TO 33Ø
34Ø  LET Y$=INKEY$
35Ø  IF Y$="X" THEN GO SUB 1ØØØ
353  IF Y$="S" THEN GO SUB 2ØØØ
355  IF Y$="Z" THEN GO TO 3ØØØ
36Ø  FOR I=1 TO 15
37Ø  IF Y$=X$(I) THEN GO TO 4ØØ
38Ø  NEXT I
39Ø  GO TO 33Ø
4ØØ  BEEP .25,N(I) : BORDER C(I)
41Ø  GO TO 33Ø
1ØØØ LET S$="33333335123_44443333355421"
1Ø1Ø FOR J=1 TO LEN S$
1Ø2Ø FOR K=1 TO 1Ø
```

```
1Ø3Ø  IF S$(J) =X$(K) THEN GO TO 1Ø5Ø
1Ø4Ø  NEXT K
1Ø45  GO TO 1Ø6Ø
1Ø5Ø  BEEP .25,N(K) : BORDER C(K)
1Ø55  PAUSE 5
1Ø6Ø  NEXT J
1Ø7Ø  RETURN
2ØØØ  FOR I=1 TO 15
2Ø1Ø  BEEP .25,N(I) : BORDER C(I) : PAUSE 5
2Ø2Ø  NEXT I
2Ø3Ø  FOR I=15 TO 1 STEP -1
2Ø4Ø  BEEP .25,N(I) : BORDER C(I) : PAUSE 5
2Ø5Ø  NEXT I
2Ø6Ø  RETURN
3ØØØ  BORDER 7
3Ø1Ø  STOP
```

Points of Interest

Lines 100–180: These lines display the title.

Lines 190–200: These lines supply the dimension for the necessary arrays.

Lines 220–270: These lines draw the keys of the keyboard.

Lines 275–280: These lines set up the names of the keys.

Lines 290–305: These lines initialize the necessary arrays and print out the key labels.

Line 315: This line prints the user instructions.

Line 320: This line enters the data for the various notes.

Lines 330–355: These lines control the procedure for reading the keyboard.

Lines 360–380: These lines check that a valid key was entered.

Line 400: This line plays the note that was entered and checks the color of the border.

Line 1000: This line sets up a string of the keys for a particular tune.

Lines 1010–1070: These lines actually play the tune set up in line 1000.

Lines 2000–2060: These lines play a scale of the available notes in ascending and descending order.

Lines 3000–3010: These lines stop the program.

Possible Adaptations

1. The tune in line 1000 could easily be changed.
2. If the row of keys is not convenient, change line 280 to any desired string of 15 characters.
3. For the more musically inclined, a scale of half notes would probably be more desirable. However, to incorporate half tones it may be necessary to restrict the program to one and a half octaves.

Chapter 27

NIGHTFALL

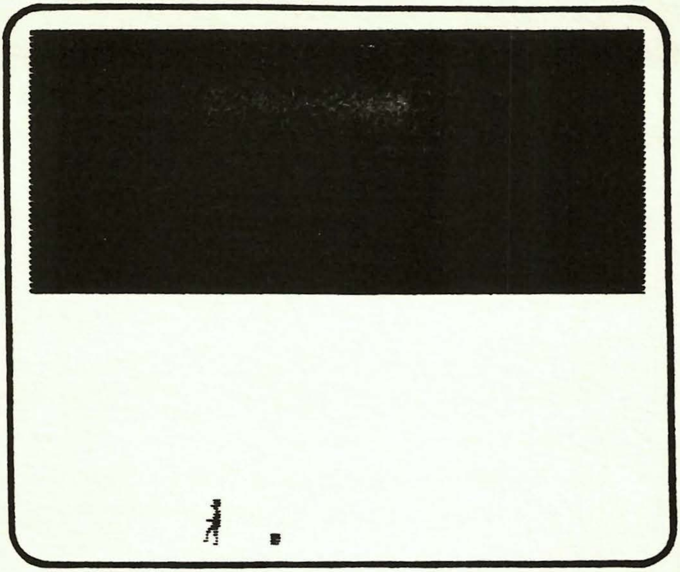
- The Game
- Program 27: NIGHTFALL
- Points of Interest
- Possible Adaptations

The Game

In this game, you have to assist the little man at the bottom of the screen to get home before night falls. He has chosen a short cut through a park where the path is strewn with rocks. Your job is to make him jump over each rock as it is encountered. You do this by hitting the J key just before he hits the rock.

The little man runs at two different speeds. Initially the speed is set to slow, but if an obstacle is successfully avoided he is immediately switched to fast. However, if he stumbles over a rock, he is injured and returns to the slow speed. If the little man succeeds in avoiding an obstacle while in slow speed, you gain 500 points for your efforts. If he avoids an obstacle while in fast speed, you gain 1000 points. However, if the man gets injured by an obstacle in either mode, you lose 100 points.

Meanwhile, night is falling at one of two different rates, depending upon the speed at which the little man is running. If he is in slow mode, night falls faster, and if he is in fast mode, night falls slower. During any given right-to-left pass across the screen, only one jump is permitted. Once night falls (a dark veil eventually covers the whole screen), the game is over, and the final score is displayed in flashing mode.



Program 27: NIGHTFALL

```
100  CLS
110  PRINT AT 8,11;"NIGHT_FALL"
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
130  PRINT AT 13,13;"(c)-1984"
140  FOR I=1 TO 3
150  FOR J=0 TO 6
160  BORDER J: PAUSE 20
170  NEXT J
180  NEXT I
190  BORDER 5
200  LET R=175
210  LET RK=0
220  LET F=0
```

```
230  LET W=4
500  POKE USR "A", BIN 00001110
510  POKE USR "A"+1,BIN 00001110
520  POKE USR "A"+2,BIN 00011110
530  POKE USR "A"+3,BIN 00001100
540  POKE USR "A"+4,BIN 00000110
550  POKE USR "A"+5,BIN 00001110
560  POKE USR "A"+6,BIN 00001110
570  POKE USR "A"+7,BIN 01111110
600  POKE USR "B",BIN 00001110
610  POKE USR "B"+1,BIN 00011110
620  POKE USR "B"+2,BIN 00111100
630  POKE USR "B"+3,BIN 01101100
640  POKE USR "B"+4,BIN 01001100
650  POKE USR "B"+5,BIN 01000100
660  POKE USR "B"+6,BIN 11000100
670  POKE USR "B"+7,BIN 00001100
700  POKE USR "C", BIN 00111110
710  POKE USR "C"+1,BIN 11111110
720  POKE USR "C"+2,BIN 11001100
730  POKE USR "C"+3,BIN 10001110
740  POKE USR "C"+4,BIN 10000111
750  POKE USR "C"+5,BIN 00000011
760  POKE USR "C"+6,BIN 00000001
770  POKE USR "C"+7,BIN 00000000
775  REM ************************
780  REM *    - GRAPHICS A    *
785  REM *    - GRAPHICS B    *
790  REM *    - GRAPHICS C    *
795  REM ************************
```

```
 800  PAPER 6
 805  CLS
 810  LET S$="________________"
 820  FOR I=0 TO 21
 830  IF I=19 THEN BRIGHT 1
 840  PRINT AT I,0;S$;S$
 850  NEXT I
1000  LET SCR=0
1005  LET X=31 : LET SPD=1
1010  PRINT AT 20,X;"_"; AT 21,X;"_"
1015  LET X=31
1016  PRINT AT 21,RK;"_"
1017  GO SUB 4000
1020  PRINT AT 20,X;"[graphic]";AT 21,X;"[graphic]"
1025  GO SUB 3000
1030  IF X-SPD<0 THEN GO TO 1010
1035  LET F=0
1040  LET Y=X
1045  LET Z=Y-(SPD AND 2)
1050  LET X=X-SPD
1052  IF INKEY$="J" AND HIT=0 AND RK< >X AND
      RK< >X-1 THEN GO SUB 6000
1055  IF X=RK OR X-1=RK THEN GO SUB 5000
1060  IF F=1 THEN GO TO 1030
1065  GO SUB 2000
1070  GO TO 1030
2000  PRINT AT 20,Y;"_";AT 21,Y;"_"
2005  PAUSE 2
2010  PRINT AT 20,Z;"[graphic]";AT 21,Z;"[graphic]"
2015  PAUSE 2
```

```
2Ø2Ø  PRINT AT 2Ø,Z;"_";AT 21,Z;"_"
2Ø3Ø  PRINT AT 2Ø,X;"[graphic]";AT 21,X;"[graphic]"
2Ø4Ø  RETURN
3ØØØ  FOR K=R TO R-W STEP -1
3Ø1Ø  PLOT Ø,K
3Ø2Ø  DRAW 255,Ø
3Ø3Ø  NEXT K
3Ø4Ø  BEEP .25,8: PAUSE 5: BEEP .25,1Ø
3Ø5Ø  LET R=R-(W+1)
3Ø6Ø  IF R<=23 THEN GO TO 9ØØØ
3Ø7Ø  RETURN
4ØØØ  LET RK=INT (RND*1Ø)*2+2
4Ø1Ø  PRINT AT 21,RK;" ■"
4Ø2Ø  LET HIT=Ø
4Ø3Ø  RETURN
5ØØØ  IF INKEY$="J" AND HIT<>1 THEN GO TO 52ØØ
5Ø1Ø  GO SUB 2ØØØ
5Ø2Ø  FLASH 1
5Ø3Ø  PRINT AT 2Ø,X;"[graphic]";AT 21,X;"[graphic]"
5Ø4Ø  FLASH Ø
5Ø5Ø  BEEP 2.5,-2Ø
5Ø6Ø  PAUSE 1Ø
5Ø7Ø  LET SPD=1
5Ø8Ø  PRINT AT 2Ø,X;"_";AT 21,X;"_"
5Ø9Ø  LET W=6
51ØØ  LET SCR=SCR-1ØØ
511Ø  LET X=RK-1
512Ø  RETURN
52ØØ  PRINT AT 2Ø,Y;"_";AT 21,Y;"_"
521Ø  PRINT AT 19,RK;"[graphic]";AT 2Ø,RK;"[graphic]"
522Ø  PAUSE 15
```

```
5230  PRINT AT 19,RK;"_";AT 20,RK;"_"
5240  PRINT AT 20,RK-1;"";AT 21,RK-1;""
5250  FOR K=1 TO 3
5260  BEEP .5,20
5270  NEXT K
5280  LET SCR=SCR+SPD*500
5290  LET SPD=2
5300  PRINT AT 21,RK;" ■"
5310  LET F=1
5320  LET W=4
5330  LET X=RK-1
5340  RETURN
6000  LET HIT=1
6010  PRINT AT 20,Y;"_";AT 21,Y;"_";AT 19,Z;"";
      AT 20,Z;""
6020  PAUSE 10: PRINT AT 19,Z;"_";AT 20,Z;"_"
6030  PRINT AT 20,X;"";AT 21,X;""
6040  PRINT AT 20,X;"_";AT 21,X;"_"
6050  LET X=X-SPD
6060  LET F=1
6070  RETURN
9000  FLASH 1
9010  PRINT AT 8,9;"NIGHT_HAS_FALLEN"
9020  PRINT AT 11,7;"FINAL_SCORE==>_";SCR
9030  FLASH 0
9040  BRIGHT 0
9050  STOP
```

Points of Interest

Lines 100–180: These lines display the title of the program.

Lines 190–230: These instructions initialize certain variables.

Lines 500–570: These lines enter the user definition of a graphic symbol representing the top half of the little man.

Lines 600–770: These lines enter the user definition of a graphic symbol for the two different lower halves of the man, giving a running effect.

Lines 800–850: These lines set up the initial screen display.

Lines 1000–1070: These lines are the control routine to display the running man and to handle the keyboard.

Lines 2000–2040: This subroutine displays the running man.

Lines 3000–3070: This subroutine displays the descending night.

Lines 4000–4030: This subroutine randomly produces the obstacle (the rock) in the shape of a little square.

Lines 5000–5120: This subroutine supplies the visual and the audio portions of the stumbling effect, decrements the score, and increases the speed of nightfall.

Lines 5200–5340: This subroutine displays the man actually jumping over the obstacle (with a sound accompaniment), increases the score according to the speed, and slows down nightfall.

Lines 6000–6070: This subroutine displays a missed jump.

Lines 9000–9050: These lines display the final score in flashing mode and terminate the program.

Possible Adaptations

1. You can easily change the score awards to suit yourself.
2. More than one obstacle per pass could be incorporated, but this is likely to be a difficult piece of programming.
3. The speed at which nightfall descends could be changed.
4. A moderate speed could be added to the fast and slow speeds.

Chapter 28

STAR BLAST

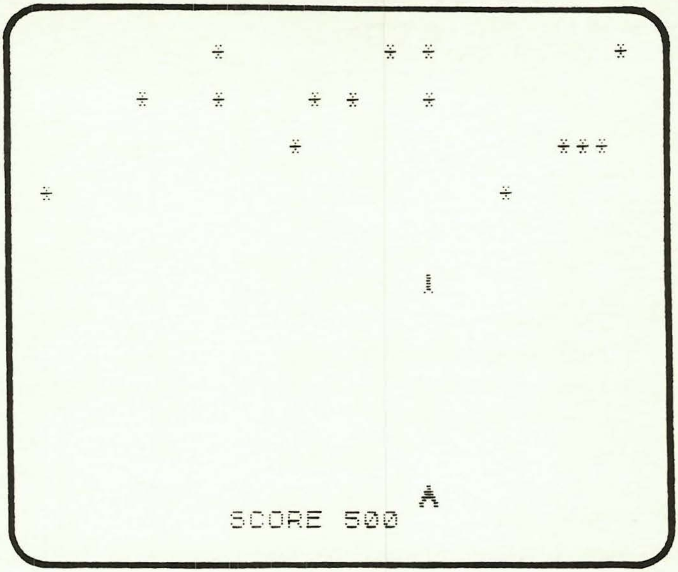
- The Game
- Program 28: STAR BLAST
- Points of Interest
- Possible Adaptations

The Game

This game is written so that either the joystick or the keyboard may be used for controlling the game. You'll find two versions of the program; choose the one designed for your equipment.

In this game, an earth station has to destroy harmful stars that block out the sun's rays. These stars move in a right-to-left direction. At the outset of the game, twenty stars are distributed in four rows at the top of the screen. A set time limit is imposed. During every 50 units of time, the stars move from the right side of the screen to the left. However, there are four different speeds involved. The top row moves at speed 1 (the slowest speed), the second row at speed 2, the third row at speed 3, and the fourth row at speed 4 (the highest speed). Using the joystick, your earth station can be moved to the left or right of the screen so that it's in a better position to shoot down the stars.

By pressing the fire button on the joystick, you may fire one missile at a time. It will either hit the star and cause it to disintegrate or harmlessly spin off the top of the screen. If a star is successfully hit, a beep is sounded, the star disappears, and you add 100 points to your score. Should you hit the fire button before a previous missile has run its course, the previous shot is terminated and the next one initiated.



If you succeed in wiping out all the stars on the screen, not only is it refilled with an additional five stars but you also have less time than before to eliminate them. No matter how good you are, you will eventually run out of time. The idea behind the game is to score as high as you can before running out of time.

With the non-joystick version, the above description applies, except that instead of using a joystick you use the keyboard in the following manner: key number 5, moves the earth station to the left, key number 8, moves the earth station to the right, and the space key is used to fire shots.

Program 28: STAR BLAST
(joystick version)

```
100  CLS
110  PRINT AT 8,11;"STAR_BLAST"
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
```

```
130  PRINT AT 13,13;"(c)-1984"
140  FOR I=1 TO 3
150  FOR J=Ø TO 6
160  BORDER J: PAUSE 2Ø
170  NEXT J
180  NEXT I
200  DIM A$(4,32)
220  LET N=15
230  LET TM=Ø
240  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
245  PAPER 1: BRIGHT 1
250  FOR J=Ø TO 21
260  PRINT AT J,Ø;S$;S$
270  NEXT J
280  LET PR=Ø
290  INK 7: PRINT AT 21,11;"SCORE _"
300  LET LIMIT=8ØØ
500  POKE USR "A", BIN ØØØ11ØØØ
510  POKE USR "A"+1,BIN ØØØ11ØØØ
520  POKE USR "A"+2,BIN ØØ1111ØØ
530  POKE USR "A"+3,BIN ØØ1111ØØ
540  POKE USR "A"+4,BIN Ø111111Ø
550  POKE USR "A"+5,BIN Ø11ØØ11Ø
560  POKE USR "A"+6,BIN 11ØØØØ11
570  POKE USR "A"+7,BIN 11ØØØØ11
600  POKE USR "B",BIN ØØØ11ØØØ
610  POKE USR "B"+1,BIN ØØØ11ØØØ
620  POKE USR "B"+2,BIN ØØØ11ØØØ
630  POKE USR "B"+3,BIN ØØØ11ØØØ
```

```
640  POKE USR "B"+4,BIN ØØØ11ØØØ
650  POKE USR "B"+5,BIN ØØØ11ØØØ
660  POKE USR "B"+6,BIN ØØ1ØØ1ØØ
670  POKE USR "B"+7,BIN ØØØØØØØØ
8ØØ  REM **********************
81Ø  REM *                    *
82Ø  REM *   A - GRAPHICS A   *
83Ø  REM *   I - GRAPHICS B   *
84Ø  REM *                    *
85Ø  REM **********************
9ØØ  LET STA=15
91Ø  FOR I=1 TO 4
93Ø  FOR J=1 TO 32
94Ø  LET A$(I,J)="_"
95Ø  NEXT J
96Ø  NEXT I
97Ø  LET SCR=Ø
98Ø  GO SUB 3ØØØ
985  GO SUB 4ØØØ
1Ø1Ø IF STICK (1,1)=4 THEN GO TO 11ØØ
1Ø2Ø IF STICK (1,1)=8 THEN GO TO 115Ø
1Ø3Ø IF STICK (2,1)=1 THEN GO TO 1Ø6Ø
1Ø5Ø GO TO 12ØØ
1Ø6Ø LET PR=1: LET S=19: LET C=STA
1Ø7Ø GO TO 12ØØ
11ØØ IF STA-1<Ø THEN GO TO 12ØØ
111Ø PRINT AT 2Ø,STA;"_"
112Ø LET STA=STA-1
113Ø GO TO 12ØØ
```

```
1150  IF STA+1>31 THEN GO TO 1200
1160  PRINT AT 20,STA;"_"
1170  LET STA=STA+1
1180  GO TO 12ØØ
12ØØ  INK 6: PRINT AT 2Ø,STA;"A"
121Ø  IF PR=Ø THEN GO TO 126Ø
122Ø  IF S<=7 AND S/2<>INT (S/2) THEN GO SUB
      6ØØØ
123Ø  INK 6: PRINT AT S,C;"I"
124Ø  BEEP .Ø1,1: PAUSE 3: PRINT AT S,C;"_"
125Ø  LET S=S-2
126Ø  LET TM=TM+1
1265  IF TM/5Ø=INT (TM/5Ø) THEN GO SUB 5ØØØ
127Ø  IF TM=LIMIT THEN GO TO 9ØØØ
128Ø  IF LEFT=Ø THEN GO TO 13ØØ
129Ø  GO TO 1Ø1Ø
13ØØ  GO SUB 3ØØØ
131Ø  GO SUB 4ØØØ
132Ø  GO TO 1Ø1Ø
3ØØØ  IF N<1ØØ THEN LET N=N+5
3ØØ5  LET LIMIT=LIMIT+5ØØ: LET LEFT=N
3Ø1Ø  FOR K=1 TO N
3Ø2Ø  LET X=INT (RND*4)+1
3Ø3Ø  LET Y=INT (RND*32)+1
3Ø4Ø  IF A$(X,Y)<>"_" THEN GO TO 3Ø2Ø
3Ø5Ø  LET A$(X,Y)="*"
3Ø6Ø  NEXT K
3Ø7Ø  RETURN
4ØØØ  FOR K=1 TO 4
```

```
4010  INK 7: PRINT AT (K-1)*2+1,0;A$(K,1 TO 32)
4020  NEXT K
4030  RETURN
5000  FOR K=1 TO 4
5005  LET H$=" "
5010  LET H$=A$(K,1 TO K)
5020  LET A$(K,1 TO 32-K)=A$(K,K+1 TO 32)
5030  LET A$(K,32-K+1 TO ) =H$
5040  NEXT K
5050  GO SUB 4000
5070  BEEP .1,-10
5080  RETURN
6000  FOR K=4 TO 1 STEP -1
6010  IF A$(K,C+1)="*" AND S=(K-1)*2+1 THEN GO
      TO 6100
6020  NEXT K
6030  IF S>0 THEN RETURN
6040  GO TO 6120
6100  LET A$(K,C+1)="_"
6103  PRINT AT (K-1)*2+1,C;"_"
6105  LET LEFT=LEFT-1
6110  LET SCR=SCR+100
6115  GO SUB 7000
6120  LET PR=0
6130  RETURN
7000  BEEP .5,30: FLASH 1
7010  PRINT AT 21,17;SCR
7020  FLASH 0
7030  RETURN
```

```
9ØØØ  BRIGHT Ø
9Ø1Ø  INK Ø
9Ø2Ø  PAPER 7
```

Program 28: STAR BLAST
(non-joystick version)

```
 1ØØ  CLS
 11Ø  PRINT AT 8,11;"STAR_BLAST"
 12Ø  PRINT AT 11,7;"Copyright-Yin_Chiu"
 13Ø  PRINT AT 13,13;"(c)-1984"
 14Ø  FOR I=1 TO 3
 15Ø  FOR J=Ø TO 6
 16Ø  BORDER J: PAUSE 2Ø
 17Ø  NEXT J
 18Ø  NEXT I
 2ØØ  DIM A$(4,32)
 22Ø  LET N=15
 23Ø  LET TM=Ø
 24Ø  LET S$="________________"
 245  PAPER 1: BRIGHT 1
 25Ø  FOR J=Ø TO 21
 26Ø  PRINT AT J,Ø;S$;S$
 27Ø  NEXT J
 28Ø  LET PR=Ø
 29Ø  INK 7: PRINT AT 21,11;"SCORE_"
 3ØØ  LET LIMIT=8ØØ
 5ØØ  POKE USR "A",BIN ØØØ11ØØØ
 51Ø  POKE USR "A"+1,BIN ØØØ11ØØØ
 52Ø  POKE USR "A"+2,BIN ØØ1111ØØ
 53Ø  POKE USR "A"+3,BIN ØØ1111ØØ
```

```
54Ø  POKE USR "A"+4,BIN Ø111111Ø
55Ø  POKE USR "A"+5,BIN Ø11ØØ11Ø
56Ø  POKE USR "A"+6,BIN 11ØØØØ11
57Ø  POKE USR "A"+7,BIN 11ØØØØ11
6ØØ  POKE USR "B",BIN ØØØ11ØØØ
61Ø  POKE USR "B"+1,BIN ØØØ11ØØØ
62Ø  POKE USR "B"+2,BIN ØØØ11ØØØ
63Ø  POKE USR "B"+3,BIN ØØØ11ØØØ
64Ø  POKE USR "B"+4,BIN ØØØ11ØØØ
65Ø  POKE USR "B"+5,BIN ØØØ11ØØØ
66Ø  POKE USR "B"+6,BIN ØØ1ØØ1ØØ
67Ø  POKE USR "B"+7,BIN ØØØØØØØØ
8ØØ  REM ***********************
81Ø  REM *                     *
82Ø  REM *      - GRAPHICS A    *
83Ø  REM *      - GRAPHICS B    *
84Ø  REM *                     *
85Ø  REM ***********************
9ØØ  LET STA=15
91Ø  FOR I=1 TO 4
93Ø  FOR J=1 TO 32
94Ø  LET A$(I,J)="_"
95Ø  NEXT J
96Ø  NEXT I
97Ø  LET SCR=Ø
98Ø  GO SUB 3ØØØ
985  GO SUB 4ØØØ
1Ø1Ø IF INKEY$="5" THEN GO TO 11ØØ
1Ø2Ø IF INKEY$="8" THEN GO TO 115Ø
```

```
1Ø3Ø  IF INKEY$="_" THEN GO TO 1Ø6Ø
1Ø5Ø  GO TO 12ØØ
1Ø6Ø  LET PR=1: LET S=19: LET C=STA
1Ø7Ø  GO TO 12ØØ
11ØØ  IF STA-1<Ø THEN GO TO 12ØØ
111Ø  PRINT AT 2Ø,STA;"_"
112Ø  LET STA=STA-1
113Ø  GO TO 12ØØ
115Ø  IF STA+1>31 THEN GO TO 12ØØ
116Ø  PRINT AT 2Ø,STA;"_"
117Ø  LET STA=STA+1
118Ø  GO TO 12ØØ
12ØØ  INK 6: PRINT AT 2Ø,STA;"A"
121Ø  IF PR=Ø THEN GO TO 126Ø
122Ø  IF S<=7 AND S/2<>INT (S/2) THEN GO SUB
      6ØØØ
123Ø  INK 6: PRINT AT S,C;"I"
124Ø  BEEP .Ø1,1: PAUSE 3: PRINT AT S,C;"_"
125Ø  LET S=S-2
126Ø  LET TM=TM+1
1265  IF TM/5Ø=INT (TM/5Ø) THEN GO SUB 5ØØØ
127Ø  IF TM=LIMIT THEN GO TO 9ØØØ
128Ø  IF LEFT=Ø THEN GO TO 13ØØ
129Ø  GO TO 1Ø1Ø
13ØØ  GO SUB 3ØØØ
131Ø  GO SUB 4ØØØ
132Ø  GO TO 1Ø1Ø
3ØØØ  IF N<1ØØ THEN LET N=N+5
3ØØ5  LET LIMIT=LIMIT+5ØØ: LET LEFT=N
```

```
3Ø1Ø  FOR K=1 TO N
3Ø2Ø  LET X=INT (RND*4)+1
3Ø3Ø  LET Y=INT (RND*32)+1
3Ø4Ø  IF A$(X,Y)<>"_" THEN GO TO 3Ø2Ø
3Ø5Ø  LET A$(X,Y)="*"
3Ø6Ø  NEXT K
3Ø7Ø  RETURN
4ØØØ  FOR K=1 TO 4
4Ø1Ø  INK 7: PRINT AT (K-1)*2+1,Ø;A$(K,1 TO 32)
4Ø2Ø  NEXT K
4Ø3Ø  RETURN
5ØØØ  FOR K=1 TO 4
5ØØ5  LET H$=" "
5Ø1Ø  LET H$=A$(K,1 TO K)
5Ø2Ø  LET A$(K,1 TO 32-K)=A$(K,K+1 TO 32)
5Ø3Ø  LET A$(K,32-K+1 TO )=H$
5Ø4Ø  NEXT K
5Ø5Ø  GO SUB 4ØØØ
5Ø7Ø  BEEP .1,-1Ø
5Ø8Ø  RETURN
6ØØØ  FOR K=4 TO 1 STEP -1
6Ø1Ø  IF A$(K,C+1)="*" AND S=(K-1)*2+1 THEN GO
      TO 61ØØ
6Ø2Ø  NEXT K
6Ø3Ø  IF S>Ø THEN RETURN
6Ø4Ø  GO TO 612Ø
61ØØ  LET A$(K,C+1)="_"
61Ø3  PRINT AT (K-1)*2+1,C;"_"
61Ø5  LET LEFT=LEFT-1
```

```
6110  LET SCR=SCR+100
6115  GO SUB 7000
6120  LET PR=0
6130  RETURN
7000  BEEP .5,30: FLASH 1
7010  PRINT AT 21,17;SCR
7020  FLASH 0
7030  RETURN
9000  BRIGHT 0
9010  INK 0
9020  PAPER 7
```

Points of Interest

Lines 100–180: These lines display the title page.

Lines 200–300: These lines initialize the matrix and two variables.

Lines 500–570: These lines represent the user-defined graphics for the earth station.

Lines 600–670: These lines represent the user-defined graphics for the missile.

Lines 800–850: These lines are optional remark statements to describe the graphics.

Lines 900–970: These lines blank out the array A$.

Line 980: This line calls subroutine 3000 to generate random stars.

Line 985: This line calls subroutine 4000 to display stars at the top of the screen.

Lines 1010–1020: These lines check the joystick or the keyboard for left–right movement.

Line 1030: This line checks the joystick or the keyboard to determine if a missile has been fired.

Lines 1050–1320: These lines handle the movement and time-keeping chores.

Lines 3000–3070: This subroutine increments the number of stars and generates a random array of stars.

Lines 4000–4030: This subroutine displays the stars.

Lines 5000–5080: This subroutine manipulates the arrays for the four rows of stars.

Lines 6000–6130: This subroutine processes a destroyed star and increments the score accordingly.

Lines 7000–7030: This subroutine displays a flashing score.

Lines 9000–9020: These lines restore the display (INK) and background (PAPER) color, and turn off the brightness.

Possible Adaptations

1. The time limit could be changed.
2. With each round the number of stars appearing could be changed.
3. The scoring system could easily be changed.
4. A sound effect could be included to simulate an explosion when a star is hit.

Chapter 29

SUPER SKETCH

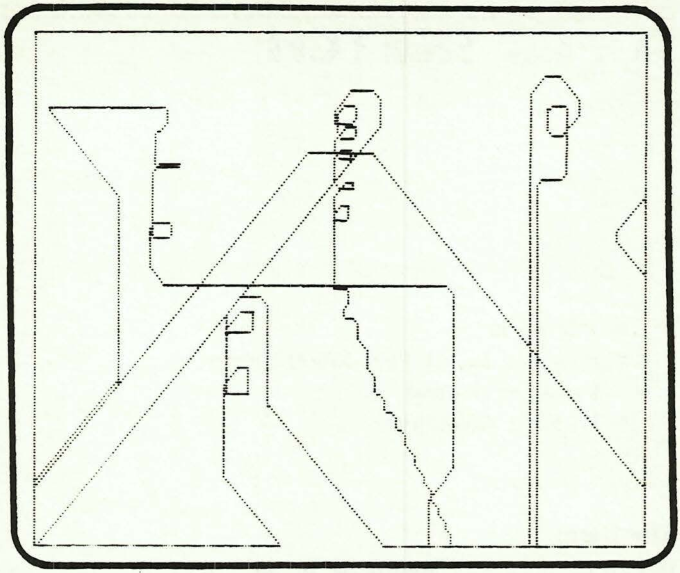
- The Game
- Program 29: SUPER SKETCH
- Points of Interest
- Possible Adaptations

The Game

If you've ever had ambitions to be a Picasso, here's your golden opportunity! Without risking paint on your clothes, you can sketch away to your heart's content. This program is given in two forms, one for use with a joystick and one with the keyboard.

When the program is run, you are invited to enter a background color: D stands for dark, and L stands for light. If L is selected, the background becomes white "paper" with black "ink." If D is selected, white "ink" is formed on a green background. After the screen is blanked out, you may begin to sketch. You always start at the lower left-hand corner of the screen. All movements you make with the joystick are reflected on the screen, and you can create quite a sophisticated sketch using the eight movements of the joystick. Don't worry about running off the edge of the screen, because bounds checking is provided for in the program.

At any point during the running of the program you may clear the screen and start again simply by pressing the button on the joystick and shifting the stick down. To stop sketching, without erasing the screen, merely press the button and shift the stick up. Should you create a design which you wish to



save, you can send a copy to a printer or to a tape; we refer you to your user manual for further information.

With the non-joystick version, use the following keys in place of the joystick control: 7 key to draw upward, 6 key to draw down, 5 key to draw to the left, and 8 key to draw to the right. For less standard moves, use the U key to move diagonally up to the right, the Y key to move diagonally up to the left, H to move diagonally down to the left, J to move diagonally down to the right, Q followed by the C key clears the screen, and Q followed by the S key stops the program.

Program 29: SUPER SKETCH (joystick version)

```
100  CLS
110  PRINT AT 8,10;"SUPER_SKETCH"
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
130  PRINT AT 13,13;"(c)-1984"
```

```
140  FOR I=1 TO 2
150  FOR J=Ø TO 6
160  BORDER J: PAUSE 2Ø
170  NEXT J
180  NEXT I
190  CLS
2ØØ  PRINT AT 3,2;"ENTER_YOUR_BACKGROUND
     _CHOICE"
21Ø  PRINT AT 5,1Ø;"D-FOR_DARK";AT 6,1Ø;"L-
     FOR_LIGHT"
22Ø  IF INKEY$=" " THEN GO TO 22Ø
23Ø  IF INKEY$="D" THEN GO TO 3ØØ
24Ø  IF INKEY$="L" THEN GO TO 35Ø
25Ø  GO TO 21Ø
3ØØ  PAPER 4
31Ø  INK 7
32Ø  GO TO 4ØØ
35Ø  PAPER 7
36Ø  INK Ø
4ØØ  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
41Ø  FOR I=Ø TO 21
42Ø  PRINT AT I,Ø;S$;S$
43Ø  NEXT I
45Ø  LET R=Ø
46Ø  LET C=Ø
5ØØ  LET X= STICK (1,1)
51Ø  IF STICK (2,1)=1 THEN GO TO 3ØØØ
52Ø  IF X=Ø THEN GO TO 5ØØ
53Ø  GO TO 9ØØ+X*1ØØ
```

```
1ØØØ  IF R+1>175 THEN GO TO 5ØØ
1Ø1Ø  LET R=R+1
1Ø2Ø  DRAW Ø,1
1Ø3Ø  GO TO 5ØØ
11ØØ  IF R-1<Ø THEN GO TO 5ØØ
111Ø  LET R=R-1
112Ø  DRAW Ø,-1
113Ø  GO TO 5ØØ
13ØØ  IF C-1<Ø THEN GO TO 5ØØ
131Ø  LET C=C-1
132Ø  DRAW -1,Ø
133Ø  GO TO 5ØØ
14ØØ  IF R+1>175 OR C-1<Ø THEN GO TO 5ØØ
141Ø  LET R=R+1
142Ø  LET C=C-1
143Ø  DRAW -1,1
144Ø  GO TO 5ØØ
15ØØ  IF R-1<Ø OR C-1<Ø THEN GO TO 5ØØ
151Ø  LET R=R-1
152Ø  LET C=C-1
153Ø  DRAW -1,-1
154Ø  GO TO 5ØØ
17ØØ  IF C+1>255 THEN GO TO 5ØØ
171Ø  LET C=C+1
172Ø  DRAW 1,Ø
173Ø  GO TO 5ØØ
18ØØ  IF R+1>175 OR C+1>255 THEN GO TO 5ØØ
181Ø  LET R=R+1
182Ø  LET C=C+1
183Ø  DRAW 1,1
```

```
1840  GO TO 500
1900  IF R-1<0 OR C+1>255 THEN GO TO 500
1910  LET R=R-1
1920  LET C=C+1
1930  DRAW 1,-1
1940  GO TO 500
3000  LET Z= STICK (1,1)
3010  IF Z=1 THEN STOP
3020  IF Z=2 THEN GO TO 3040
3030  GO TO 3000
3040  CLS
3050  GO TO 410
```

Program 29: SUPER SKETCH
(non-joystick version)

```
 100  CLS
 110  PRINT AT 8,10;"SUPER_SKETCH"
 120  PRINT AT 11,7;"Copyright-Yin_Chiu"
 130  PRINT AT 13,13;"(c)-1984"
 140  FOR I=1 TO 2
 150  FOR J=0 TO 6
 160  BORDER J: PAUSE 20
 170  NEXT J
 180  NEXT I
 190  CLS
 200  PRINT AT 3,2;"ENTER_YOUR_BACKGROUND
      _CHOICE"
 210  PRINT AT 5,10;"D-FOR_DARK";AT 6,10;"L-
      FOR_LIGHT"
 220  IF INKEY$="" THEN GO TO 220
```

```
230  IF INKEY$="D" THEN GO TO 300
240  IF INKEY$="L" THEN GO TO 350
250  GO TO 210
300  PAPER 4
310  INK 7
320  GO TO 400
350  PAPER 7
360  INK 0
400  LET S$="________________"
410  FOR I=0 TO 21
420  PRINT AT I,0;S$;S$
430  NEXT I
450  LET R=0
460  LET C=0
500  LET X$=INKEY$
510  IF X$="Q" THEN GO TO 3000
520  IF X$="" THEN GO TO 500
530  IF X$="7" THEN GO TO 1000
540  IF X$="6" THEN GO TO 1100
550  IF X$="8" THEN GO TO 1700
560  IF X$="5" THEN GO TO 1300
570  IF X$="U" THEN GO TO 1800
580  IF X$="Y" THEN GO TO 1400
590  IF X$="J" THEN GO TO 1900
600  IF X$="H" THEN GO TO 1500
610  GO TO 500
1000 IF R+1>175 THEN GO TO 500
1010 LET R=R+1
```

```
1020  DRAW 0,1
1030  GO TO 500
1100  IF R-1<0 THEN GO TO 500
1110  LET R=R-1
1120  DRAW 0,-1
1130  GO TO 500
1300  IF C-1<0 THEN GO TO 500
1310  LET C=C-1
1320  DRAW -1,0
1330  GO TO 500
1400  IF R+1>175 OR C-1<0 THEN GO TO 500
1410  LET R=R+1
1420  LET C=C-1
1430  DRAW -1,1
1440  GO TO 500
1500  IF R-1<0 OR C-1<0 THEN GO TO 500
1510  LET R=R-1
1520  LET C=C-1
1530  DRAW -1,-1
1540  GO TO 500
1700  IF C+1>255 THEN GO TO 500
1710  LET C=C+1
1720  DRAW 1,0
1730  GO TO 500
1800  IF R+1>175 OR C+1>255 THEN GO TO 500
1810  LET R=R+1
1820  LET C=C+1
1830  DRAW 1,1
1840  GO TO 500
```

```
1900  IF R-1<0 OR C+1>255 THEN GO TO 500
1910  LET R=R-1
1920  LET C=C+1
1930  DRAW 1,-1
1940  GO TO 500
3000  LET X$=INKEY$
3010  IF X$="S" THEN STOP
3020  IF X$="C" THEN GO TO 3040
3030  GO TO 3000
3040  CLS
3050  GO TO 410
```

Points of Interest

Lines 100–180: These lines display the title.

Lines 200–460: These lines handle the background (PAPER) and display (INK) color.

Lines 500–530: In the joystick version, these lines function as a control routine to process the joystick movements.

Lines 500–610: In the non-joystick version, these lines handle the keyboard for drawing in eight different directions.

Lines 1000–1030: These lines process the upward movement of the joystick.

Lines 1100–1130: These lines process the downward movement of the joystick.

Lines 1300–1330: These lines process the leftward movement of the joystick.

Lines 1400–1440: These lines process the diagonal movement of the joystick toward the upper left.

Lines 1500–1540: These lines process the diagonal movement of the joystick toward the lower left.

Lines 1700–1730: These lines process the rightward movement of the joystick.

Lines 1800–1840: These lines process the diagonal movement of the joystick toward the upper right.

Lines 1900–1940: These lines process the diagonal movement of the joystick toward the lower right.

Lines 3000–3050: These lines represent the routine to handle control if the joystick button is pressed. In the non-joystick version, these lines clear the screen or stop the program in response to the appropriate keys.

Possible Adaptations

1. The background paper and ink colors may be changed to any that you desire.
2. You may wish to change the starting position, perhaps to the center of the screen.
3. You may wish to add a beep each time a point is drawn.

Chapter 30

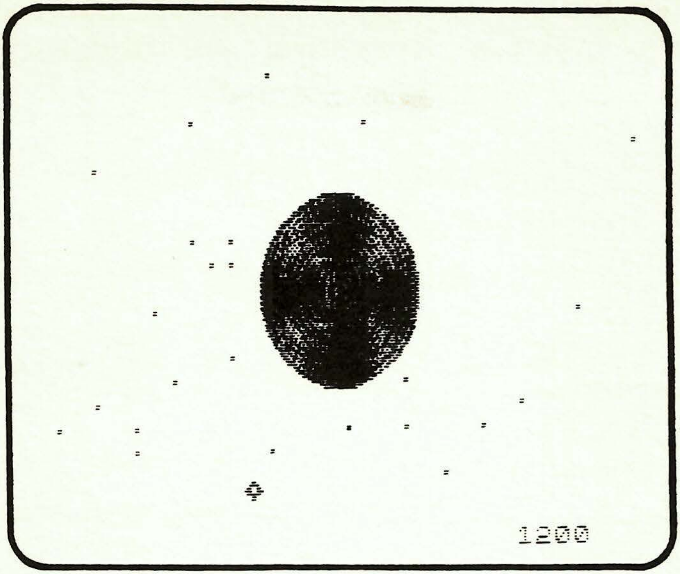
SPECTRE ZONE

- The Game
- Program 30: SPECTRE ZONE
- Points of Interest
- Possible Adaptations

The Game

As a space ranger, you have been sent to investigate the unaccountable disappearances of spaceships traveling in an uncharted sector of the galaxy. As your ship travels along the top of the screen, you suddenly encounter a mysterious force (the spectre zone, displayed as a red ball) which pulls your ship into orbit around it.

The spectre zone is a zone of energy which extracts energy from your ship and grows as a result. However, your ship's sensors detect that the forces acting on you are not evenly distributed. There are random visual breaks (specified by the appearance of a white X on the screen) that appear periodically in the path of your ship's orbit around the spectre zone. As you pass over each break point you're able to repel the spectre zone's forces and extract some of its energy by pressing any key on the keyboard. If you are successful, you not only deprive the spectre zone of some of its energy but you also slow down its growth. If you ever accumulate 2500 units of energy, you will have sufficient strength to break out of the orbit and zoom away out of the spectre zone's influence. If not, the spectre zone will engulf yet another victim—you!



Program 30: SPECTRE ZONE

```
100  CLS
110  PRINT AT 8,10;"SPECTRE_ZONE"
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
130  PRINT AT 13,13;"(c)-1984"
140  FOR I=1 TO 3
150  FOR J=0 TO 6
160  BORDER J: PAUSE 20
170  NEXT J
180  NEXT I
200  CLS
210  DIM C(32,2)
220  FOR I=1 TO 32
230  READ C(I,1),C(I,2)
```

```
240  NEXT I
250  DATA Ø,16,1,18,2,2Ø,3,22,4,23,6,24,8,25,1Ø,26,11,
     26,13,25,15,24,17,23,18,22,19,2Ø,2Ø,18
260  DATA 21,16,21,15,2Ø,13,19,11,18,9,17,8,15,7,13,
     6,11,5,1Ø,5,8,6,6,7,4,8,3,9,2,11,1,13,Ø,15
270  PAPER 1: BRIGHT 1
280  LET S$="_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
290  FOR I=Ø TO 21
300  PRINT AT I,Ø;S$;S$
310  NEXT I
320  LET SCR=Ø
330  GO SUB 1ØØØ
340  FLASH 1
350  PRINT AT 21,25;Ø
360  FLASH Ø
370  LET F1=1: LET F2=Ø
400  POKE USR "A",BIN ØØØ11ØØØ
410  POKE USR "A"+1,BIN ØØ1111ØØ
420  POKE USR "A"+2,BIN Ø11ØØ11Ø
430  POKE USR "A"+3,BIN 111ØØ111
440  POKE USR "A"+4,BIN Ø111111Ø
450  POKE USR "A"+5,BIN ØØØ11ØØØ
460  POKE USR "A"+6,BIN ØØØ11ØØØ
470  POKE USR "A"+7,BIN ØØØØØØØØ
480  REM **********************
490  REM *                    *
500  REM *  ♠ - GRAPHICS A    *
510  REM *                    *
520  REM **********************
```

```
530  FOR I=Ø TO 14
540  INK 7
550  PRINT AT Ø,I;"♠"
560  BEEP .Ø1,1Ø
570  PAUSE 5
580  PRINT AT Ø,I;"_"
590  NEXT I
600  FOR I=1 TO 73 STEP 2
610  INK 2
620  CIRCLE 128,88,I
625  CIRCLE 128,88,I+1
630  LET Q=INT (RND*2Ø)+5
640  INK 6: PRINT AT C(Q,1),C(Q,2);"X"
645  PAUSE 7: PRINT AT C(Q,1),C(Q,2);"_"
650  FOR K=1 TO 3
660  BEEP .1,25
670  NEXT K
675  LET FLAG=Ø
680  FOR J=1 TO 32
690  LET X=C(J,1)
700  LET Y=C(J,2)
710  INK 7: PRINT AT X,Y;"♠"
715  BEEP .Ø1,1Ø
720  PRINT AT X,Y;"_"
730  IF INKEY$<>"" THEN GO SUB 2ØØØ
740  NEXT J
745  IF F2=1 THEN GO TO 5ØØØ
750  NEXT I
760  GO SUB 3ØØØ
```

```
 77Ø  GO TO 4ØØØ
1ØØØ  INK 7
1ØØ5  FOR W=1 TO 25
1Ø1Ø  LET R=INT (RND*22)
1Ø2Ø  LET C=INT (RND*32)
1Ø3Ø  PRINT AT R,C;"."
1Ø4Ø  NEXT W
1Ø5Ø  RETURN
2ØØØ  IF FLAG=1 THEN RETURN
2Ø1Ø  IF J=Q THEN GO TO 21ØØ
2Ø2Ø  BEEP .5,-2Ø
2Ø3Ø  LET FLAG=1
2Ø4Ø  RETURN
21ØØ  FOR N=1 TO 4
211Ø  BEEP .1,N*1Ø
212Ø  NEXT N
213Ø  LET SCR=SCR+1ØØ
214Ø  LET I=I-1
215Ø  FLASH 1
216Ø  PRINT AT 21,25;SCR
217Ø  FLASH Ø
218Ø  IF SCR=25ØØ THEN LET F2=1
219Ø  RETURN
3ØØØ  LET Z=INT (RND*2Ø)+5
3Ø1Ø  INK 7
3Ø2Ø  FOR I=1 TO Z
3Ø3Ø  LET X=C(I,1)
3Ø4Ø  LET Y=C(I,2)
3Ø5Ø  PRINT AT X,Y;"♠"
```

```
3Ø6Ø  BEEP .Ø1,3Ø
3Ø7Ø  PRINT AT X,Y:"_"
3Ø8Ø  NEXT I
3Ø9Ø  FLASH 1
31ØØ  IF F1=1 THEN PRINT AT C(I,1),C(I,2);"♠"
311Ø  FLASH Ø
312Ø  RETURN
4ØØØ  INK 2
4Ø1Ø  FOR J=71 TO 82
4Ø2Ø  BEEP .1,-3Ø
4Ø3Ø  CIRCLE 128,88,J
4Ø4Ø  NEXT J
4Ø5Ø  FLASH 1
4Ø6Ø  INK 7
4Ø7Ø  PRINT AT 21,2;"GAME_OVER"
4Ø8Ø  FLASH Ø
4Ø9Ø  GO TO 9ØØØ
5ØØØ  LET F1=Ø
5Ø1Ø  GO SUB 3ØØØ
5Ø2Ø  LET S1=C(Z,1)
5Ø3Ø  LET S2=C(Z,2)
5Ø4Ø  IF C(Z,2)>16 THEN GO TO 51ØØ
5Ø5Ø  LET S3=Ø
5Ø6Ø  LET S4=-1
5Ø7Ø  GO TO 52ØØ
51ØØ  LET S3=31
511Ø  LET S4=1
52ØØ  INK 7
521Ø  FOR K=S2 TO S3 STEP S4
```

```
5215  IF K=S3 THEN FLASH 1
522Ø  PRINT AT S1,K;"♠"
523Ø  BEEP .Ø1,25: BEEP .Ø1,35
524Ø  IF K<>S3 THEN PRINT AT S1,K;"_"
525Ø  NEXT K
9ØØØ  PAPER 7
9Ø1Ø  INK Ø
9Ø2Ø  BRIGHT Ø
9Ø3Ø  FLASH Ø
9Ø4Ø  STOP
```

Points of Interest

Lines 100–180: These lines display the title for the program.

Lines 200–260: These lines set up your ship's orbital path.

Lines 270–370: These lines initialize the screen with the stars.

Lines 400–470: These lines create a user-defined graphics symbol for the spaceship.

Lines 480–520: These lines are optional remark statements which describe the graphic symbol.

Lines 530–590: These lines display the initial path of the spaceship.

Lines 600–770: These lines represent the control routine for the growing spectre zone. They also check the keyboard and display the zone break.

Lines 1000–1050: This subroutine displays the stars in the spectre zone.

Lines 2000–2040: These lines handle an unsuccessful attempt at extracting energy from the spectre zone.

Lines 2100–2190: These lines handle a successful attempt at extracting energy from the spectre zone and suppress the zone growth.

Lines 3000–3120: This subroutine prints the final semi-orbit of the ship.

Lines 4000–4090: These lines display the engulfing of the ship by the spectre zone.

Lines 5000–5250: These lines display the ship's breakaway from the spectre zone.

Lines 9000–9040: These lines restore the background (PAPER) and display (INK), turn off the brightness and flashing and then terminate the program.

Possible Adaptations

1. The indication of the break point (where the X appears) flashes fairly quickly. If this proves to be too difficult to use, you may wish to retain it a little longer on the screen.
2. You might want to implement a warning sound as the spectre zone reaches its maximum size.
3. If the game seems to be too difficult, you may wish to change the energy level required for breaking away from the spectre zone.

Chapter 31

CROSSFIRE

- The Game
- Program 31: CROSSFIRE
- Points of Interest
- Possbile Adaptations

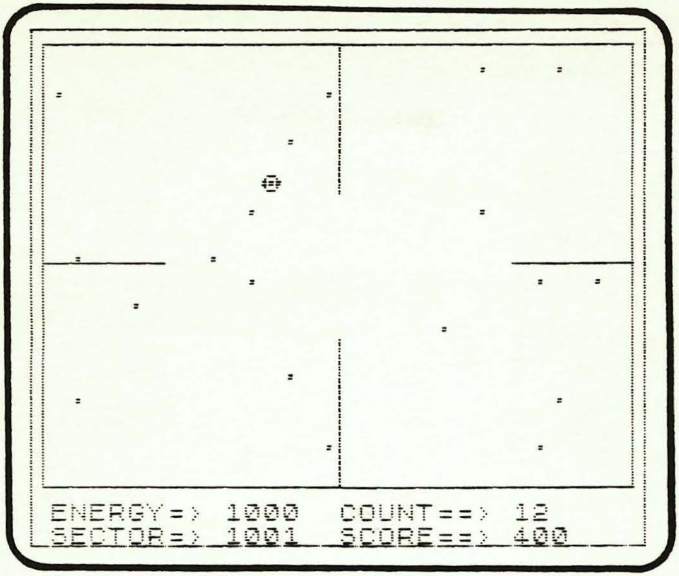
The Game

In this game, you are the pilot of a star ship under heavy attack by hostile aliens. On the screen you see a cockpit view of the sky with a grid which is the equivalent of the crosshairs of a telescopic sight. Aliens are coming at you from four different directions and at two different speeds. As they pass the center of the grid, you may press any key to fire at them. If you fail to fire at them or miss a shot, they will definitely hit you, and you will lose 500 units of shield energy. However, if you hit an alien, you receive 100 points. If you are fortunate enough to wipe out all the foreign aliens in a given sector, you gain 100 units of shield energy.

The aliens attack you one at a time. After all of them have had an opportunity to attack you from a given sector, you move on to a different sector, as shown by your sector indicator and by the pattern of the stars. Each sector has more alien ships in it than the previous one.

Program 31: CROSSFIRE

```
100  CLS
110  PRINT AT 8,12;"CROSSFIRE"
```




```
120  PRINT AT 11,7;"Copyright-Yin_Chiu"
130  PRINT AT 13,13;"(c)-1984"
140  FOR I=1 TO 3
150  FOR J=Ø TO 6
160  BORDER J: PAUSE 2Ø
170  NEXT J
180  NEXT I
200  DIM S(2Ø,2)
210  PAPER 1: BRIGHT 1
220  LET S$="________________"
230  FOR I=Ø TO 19
240  PRINT AT I,Ø;S$;S$
250  NEXT I
260  PAPER 2
```

```
270  FOR I=20 TO 21
280  PRINT AT I,Ø;S$;S$
290  NEXT I
300  GO SUB 1ØØØ
310  GO SUB 11ØØ
315  PAPER 2
320  PRINT AT 2Ø,1;"ENERGY=>";TAB
     16;"COUNT==>"
330  PRINT AT 21,1;"SECTOR=>";TAB 16;
     "SCORE==>"
340  LET SCR=Ø: LET CT=1Ø
350  LET E=5ØØØ: LET SCT=1ØØØ
355  LET OLD=CT
360  GO SUB 2ØØØ
400  POKE USR "A",BIN ØØØØØØØØ
410  POKE USR "A"+1,BIN ØØ1111ØØ
420  POKE USR "A"+2,BIN Ø1ØØØØ1Ø
430  POKE USR "A"+3,BIN 11Ø11Ø11
440  POKE USR "A"+4,BIN 11Ø11Ø11
450  POKE USR "A"+5,BIN Ø1ØØØØ1Ø
460  POKE USR "A"+6,BIN ØØ1111ØØ
470  POKE USR "A"+7,BIN ØØØØØØØØ
480  REM ***********************
482  REM *                     *
484  REM *  ⊖ - GRAPHICS A     *
486  REM *                     *
488  REM ***********************
500  FOR I=1 TO CT
510  GO SUB 3ØØØ
```

```
520 IF E<=Ø THEN GO TO 5ØØØ
530 GO SUB 2ØØØ
540 NEXT I
545 IF CT=Ø THEN LET E=E+1ØØ
550 LET CT=OLD+2
560 LET OLD=CT
570 LET SCT=SCT+1
580 FLASH 1
585 PAPER 1
590 PRINT AT 9,15;"NEW";AT 1Ø,13;"SECTOR"
600 PAUSE 35
610 FLASH Ø
620 PRINT AT 9,15;S$( TO 3);AT 1Ø,13;S$( TO 6)
630 FOR J=1 TO 7
640 BEEP .1,3Ø
645 PAUSE 3
650 NEXT J
660 GO SUB 45ØØ
670 GO TO 5ØØ
1ØØØ INK 7: PLOT 5,2Ø
1Ø1Ø DRAW 245,Ø
1Ø2Ø DRAW Ø,15Ø
1Ø3Ø DRAW -245,Ø
1Ø4Ø DRAW Ø,-15Ø
1Ø5Ø RETURN
11ØØ PAPER 1
11Ø5 FOR I=1 TO 2Ø
111Ø LET X=INT (RND*18)+1
112Ø LET Y=INT (RND*3Ø)+1
```

```
1130  PRINT AT X,Y;"."
1140  LET S(I,1) =X: LET S(I,2)=Y
1150  NEXT I
1160  RETURN
2000  INK 7: PAPER 2
2005  PLOT 128,20
2010  DRAW 0,50
2015  PLOT 128,170
2020  DRAW 0,-50
2025  PLOT 5,96
2030  DRAW 50,0
2040  PLOT 250,96
2045  DRAW -50,0
2050  FLASH 1
2060  PRINT AT 20,10;"____";AT 20,10;E
2065  PRINT AT 20,25;"__";AT 20,25;CT
2070  PRINT AT 21,10;"____";AT 21,10;SCT
2075  PRINT AT 21,25;"_____";AT 21,25;SCR
2080  FLASH 0
2090  RETURN
3000  LET X=INT (RND*2)+1
3010  LET Y=INT (RND*2)+1
3015  LET FLAG=0: INK 6
3016  PAPER 1: LET HIT=0
3020  IF X=1 THEN GO TO 3060
3030  LET P1=1: LET P2=18
3040  LET P3=INT (RND*2)+1
3050  GO TO 3100
3060  LET P1=18: LET P2=1
```

```
3070  LET P3=-(INT (RND*2)+1)
3100  IF Y=1 THEN GO TO 3160
3110  LET P4=7
3120  LET P5=ABS (P3)
3130  GO TO 3200
3160  LET P4=24
3170  LET P5=-(ABS (P3))
3200  FOR K=P1 TO P2 STEP P3
3210  PRINT AT K,P4;"⊖"
3220  BEEP .01,10
3230  PRINT AT K,P4;"_"
3240  IF INKEY$<>"" AND FLAG=0 THEN GO SUB
      4000
3245  IF HIT=1 THEN GO TO 3300
3250  LET P4=P4+P5
3260  NEXT K
3270  LET E=E-500
3280  BEEP 1,-10
3300  RETURN
4000  IF K=9 OR K=10 THEN GO TO 4100
4010  BEEP .2,-20
4020  GO TO 4150
4100  LET SCR=SCR+100
4110  FOR J=10 TO 30 STEP 10
4120  BEEP .1,J: BEEP .2,J-15
4130  NEXT J
4140  LET HIT=1
4145  LET CT=CT-1
4150  LET FLAG=1
```

```
4160  RETURN
4500  PAPER 1
4505  FOR N=1 TO 20
4510  PRINT AT S(N,1),S(N,2);"_"
4520  NEXT N
4530  GO SUB 1100
4540  GO SUB 2000
4550  RETURN
5000  FLASH 1
5010  PRINT AT 9,10;"OUT_OF_ENERGY"
5020  GO SUB 2000
9000  INK 0
9010  PAPER 7
9020  BRIGHT 0
9030  FLASH 0
9040  STOP
```

Points of Interest

Lines 100–180: These lines set up the title to the program.

Line 200: This line reserves room for the matrix S which is used for storing the locations of the stars.

Lines 210–250: These lines set up the first nineteen lines of the screen display.

Lines 260–290: These lines set up the last two lines of the screen display.

Lines 300–360: These lines initialize and display all the indicators.

Lines 400–470: These lines describe the user-defined graphics symbol for an alien vessel.

Lines 480–488: These lines are optional remark statements to describe the graphic symbols.

Lines 500–540: These lines represent the control routine for the attacking aliens.

Lines 545–670: These lines control the transition from one sector to the next.

Lines 1000–1050: This subroutine draws the grid.

Lines 1100–1160: This subroutine displays the stars in a given sector and stores their locations.

Lines 2000–2090: This subroutine displays the crosshairs and updates all the indicators.

Lines 3000–3300: This subroutine generates one of the eight possible alien moves and determines whether or not an alien has been hit. It also decrements your energy level if you miss and it sounds a hit on your starship.

Lines 4000–4160: This subroutine increments your score and sounds a tune if you are successful in destroying an alien vessel.

Lines 4500–4550: This subroutine relocates the stars in a given sector.

Lines 5000–5020: These lines display the out-of-energy condition.

Lines 9000–9040: These lines restore the display (INK) and background (PAPER), turn off the brightness and flashing and terminate the program.

Possible Adaptations

1. You may wish to change the shape of the alien vessels, or even design more than one shape for them.
2. If the game is too short, you may wish to change the number of units of shield energy to be deducted from 500 to 200 or even 100 points.

Chapter 32

SUPER SLOT II

- The Game
- Program 32: SUPER SLOT II
- Points of Interest
- Possible Adaptations

The Game

This game is a close simulation of the famous casino slot machine, sometimes known as the "one-armed bandit." It has red cherries, blue bells, purple plums, black bars, and green sevens. It also produces a payout display into which all the coins fall when you win.

The rules of the game are as follows: you start out with the sum of $50 and are allowed to bet from 1 to 4 dollars as long as there is enough in the kitty to cover the bet; a bet of zero stops the game. Once a bet has been accepted, you will see each window spin from left to right until they come to a stop. The payout is determined by the following criteria:

A leftmost cherry wins the amount of the bet times 2.

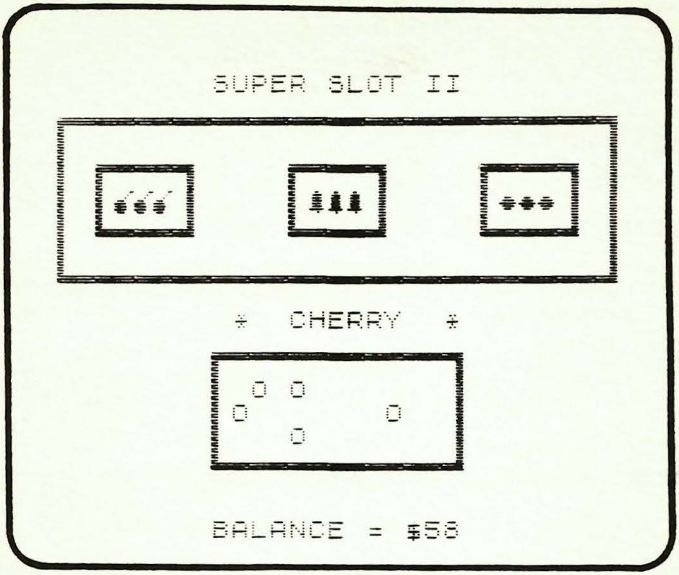
Two of a kind (left to right) wins the bet times 3.

Three of a kind (jackpot) wins the bet times 4.

Three sevens (sometimes known as super sevens) wins the bet times 7.

Program 32: SUPER SLOT II

```
100 CLS
110 PRINT AT 8,10;"SUPER_SLOT_II"
```




```
12Ø  PRINT AT 11,7;"Copyright-Yin_Chiu"
13Ø  PRINT AT 13,13;"(c)-1984"
14Ø  FOR I=1 TO 3
15Ø  FOR J=Ø TO 6
16Ø  BORDER J: PAUSE 2Ø
17Ø  NEXT J
18Ø  NEXT I
19Ø  BRIGHT 1: BORDER 2
2ØØ  PAPER 6
22Ø  LET S$="________________"
23Ø  FOR I=Ø TO 21
24Ø  PRINT AT I,Ø;S$;S$
25Ø  NEXT I
26Ø  FLASH 1
```

```
270  PAPER 2: INK 7
280  PRINT AT 2,10;"SUPER_SLOT_II"
290  FLASH 0
300  PAPER 6
400  POKE USR "A",BIN 00000011
410  POKE USR "A"+1,BIN 00000100
420  POKE USR "A"+2,BIN 00001000
430  POKE USR "A"+3,BIN 01110000
440  POKE USR "A"+4,BIN 11111000
450  POKE USR "A"+5,BIN 11111000
460  POKE USR "A"+6,BIN 11111000
470  POKE USR "A"+7,BIN 01110000
500  POKE USR "B",BIN 00011000
510  POKE USR "B"+1,BIN 00111100
520  POKE USR "B"+2,BIN 00111100
530  POKE USR "B"+3,BIN 00111100
540  POKE USR "B"+4,BIN 00111100
550  POKE USR "B"+5,BIN 00111100
560  POKE USR "B"+6,BIN 01111110
570  POKE USR "B"+7,BIN 00001000
600  POKE USR "C",BIN 00000000
610  POKE USR "C"+1,BIN 00000000
620  POKE USR "C"+2,BIN 00111000
630  POKE USR "C"+3,BIN 01111100
640  POKE USR "C"+4,BIN 11111110
650  POKE USR "C"+5,BIN 01111100
660  POKE USR "C"+6,BIN 00111000
670  POKE USR "C"+7,BIN 00000000
680  REM
```

```
700  REM ***********************
710  REM *                     *
720  REM *  🍒 - GRAPHICS A    *
730  REM *  🔔 - GRAPHICS B    *
740  REM *  ● - GRAPHICS C     *
750  REM *                     *
760  REM ***********************
770  REM
800  LET X$="BAR🔔🔔🔔🍒🍒🍒●●●*7*"
810  LET BAL=5Ø
820  LET X=16: LET Y=144
830  LET N1=232: LET N2=56
840  GO SUB 1ØØØ
850  FOR I=32 TO 192 STEP 8Ø
860  LET X=I: LET Y=128
870  LET N1=4Ø: LET N2=24
880  GO SUB 1ØØØ
890  NEXT I
900  LET X=8Ø: LET Y=64
910  LET N1=1Ø4: LET N2=4Ø
920  GO SUB 1ØØØ
930  GO TO 2ØØØ
1ØØØ INK 4
1Ø1Ø FOR K=1 TO 3
1Ø2Ø PLOT X,Y
1Ø3Ø DRAW N1,Ø: DRAW Ø,-N2
1Ø4Ø DRAW -N1,Ø: DRAW Ø,N2
1Ø5Ø LET X=X+1: LET Y=Y-1
1Ø6Ø LET N1=N1-2: LET N2=N2-2
```

```
1070  NEXT K
1080  RETURN
2000  GO SUB 4000
2010  LET C=5
2020  GO SUB 3000
2030  LET Q1=Z
2040  LET C=15
2050  GO SUB 3000
2060  LET Q2=Z
2070  LET C=25
2080  GO SUB 3000
2090  LET Q3=Z
2100  IF Q1=Q2 AND Q2=Q3 THEN GO TO 2500
2110  IF Q1=Q2 THEN GO TO 2600
2120  IF Q1=3 THEN GO TO 2700
2130  INK 0: PAPER 6
2140  FLASH 1
2150  PRINT AT 12,13;"YOU_LOSE"
2160  BEEP .75,-10
2170  LET F=-1
2180  GO TO 2800
2500  IF Q1=5 THEN GO TO 2550
2510  INK 2: PAPER 7: FLASH 1
2520  PRINT AT 12,12;"*_JACKPOT_*"
2530  LET F=4
2540  GO TO 2800
2550  INK 2: PAPER 7: FLASH 1
2560  FOR H=1 TO 7
2570  BEEP .1,25
```

```
2580  NEXT H
2590  PRINT AT 12,7;"*_SUPER_7_JACKPOT_*"
2592  LET F=7
2594  GO TO 28ØØ
26ØØ  FLASH 1: INK 2: PAPER 7
261Ø  PRINT AT 12,11;"*_2_MATCH_*"
262Ø  LET F=3
263Ø  GO TO 28ØØ
27ØØ  INK 2: PAPER 7: FLASH 1
271Ø  PRINT AT 12,11;"*_ _CHERRY_ _*"
272Ø  LET F=2
28ØØ  IF F<Ø THEN GO TO 29ØØ
281Ø  FOR H=1Ø TO 4Ø STEP 1Ø
282Ø  BEEP .2,H: BEEP .3,H-5
283Ø  NEXT H
284Ø  GO SUB 5ØØØ
29ØØ  LET BAL=BAL+F*B
291Ø  INK Ø: PAPER 6
292Ø  PAUSE 2Ø
293Ø  FLASH Ø
2935  PRINT AT 21,Ø;S$;S$
294Ø  PRINT AT 21,1Ø;"BALANCE_=_$";BAL
2944  IF BAL<=Ø THEN GO TO 8ØØØ
2946  GO SUB 4ØØØ
295Ø  PRINT AT 7,5;S$( TO 3);AT 7,15;S$( TO 3); AT
      7,25;S$( TO 3)
296Ø  PRINT AT 12,Ø;S$;S$
297Ø  GO TO 2Ø1Ø
3ØØØ  LET Z=INT (RND*5)+1
```

```
3Ø1Ø  LET T=INT (RND*5)+2
3Ø2Ø  LET SPIN=T*5+Z
3Ø3Ø  LET G=Ø
3Ø4Ø  FOR J=1 TO 5
3Ø5Ø  INK J-1
3Ø6Ø  LET P$=X$((J-1)*3+1 TO (J-1)*3+3)
3Ø7Ø  PRINT AT 7,C;P$
3Ø8Ø  BEEP .Ø1,J+2Ø
3Ø9Ø  LET G=G+1
31ØØ  IF G=SPIN THEN RETURN
311Ø  NEXT J
312Ø  GO TO 3Ø4Ø
4ØØØ  PAPER 6: INK Ø
4Ø1Ø  PRINT AT 2Ø,12;"BET?_1_TO_4"
4Ø2Ø  INPUT B
4Ø3Ø  PRINT AT 2Ø,11;S$
4Ø4Ø  IF B>=1 AND B<=4 AND B<=BAL THEN
      RETURN
4Ø5Ø  BEEP .5,-15
4Ø6Ø  IF B=Ø THEN GO TO 8ØØØ
4Ø7Ø  IF B>BAL AND B<5 THEN GO TO 4Ø9Ø
4Ø8Ø  GO TO 4ØØØ
4Ø9Ø  PRINT AT 2Ø,3;"YOU_DO_NOT_HAVE_THAT
      _MUCH"
41ØØ  PAUSE 4Ø
411Ø  PRINT AT 2Ø,Ø;S$;S$
412Ø  GO TO 4ØØØ
5ØØØ  PAPER 6: FLASH Ø
5Ø1Ø  FOR K=1 TO F*B
```

```
5020 LET D1=INT (RND*3)+15
5030 LET D2+INT (RND*11)+11
5040 PRINT AT D1,D2;"O"
5050 BEEP .05,-20
5060 NEXT K
5070 FOR K=15 TO 17
5080 PRINT AT K,11;S$(TO 11)
5090 NEXT K
5100 RETURN
8000 PAPER 6
8005 PRINT AT 12,0;S$;S$
8010 INK 2: PAPER 7: FLASH 1
8020 PRINT AT 12,12;"GAME_OVER"
8030 FOR K=10 TO 30 STEP 10
8040 BEEP .2,K: BEEP .1,K-4: BEEP .3,K+5
8050 NEXT K
9000 PAPER 7
9010 INK 0
9020 FLASH 0
9030 BRIGHT 0
9040 STOP
```

Points of Interest

Lines 100–180: These lines display the title.

Lines 200–250: These lines set up the background color for the screen.

Lines 400–670: These lines define the three graphic symbols—cherry, bell, plum.

Lines 700–770: These are optional remark statements which describe the graphics characters.

Line 800: This line sets up a string of possible displays for the slot machine windows.

Lines 820–920: These lines act as a control team for the drawing of the three windows and the payment window.

Lines 1000–1080: This subroutine actually draws the four windows.

Lines 2000–2090: These lines represent the control routine for the processing of the three displays, one at a time, left to right.

Lines 2100–2120: These lines check to see if the bet was a winning bet.

Lines 2120–2180: These lines notify you if you have lost.

Lines 2500–2540: These lines handle the jackpot display.

Lines 2550–2594: These lines handle the super seven jackpot.

Lines 2600–2630: These lines handle a two-item match.

Lines 2700–2720: These lines handle a leftmost cherry.

Lines 2800–2970: For a winning bet, a tune is played, the payout routine is called, the balance is calculated and displayed, and the three windows are blanked out.

Lines 3000–3121: This is the routine for processing and displaying the spinning windows.

Lines 4000–4120: These lines both accept and validate the desired bet.

Lines 5000–5100: These lines display the amount paid out in the payout window.

Lines 8000–8050: These lines display "GAME OVER" and play a little melody.

Lines 9000–9040: These lines restore the background (PAPER) to white and the display (INK) to black and turn off any flashing or brightness indicators before terminating the program.

Possible Adaptations

1. The betting routine at line 4000 checks that bets are not greater than your balance, greater than $4, or less

than $1. However, it does not check that the amount betted is a whole number. It is probably a good idea to make this amendment.

2. You may wish to add to or change the graphic symbols (by defining your own or perhaps amending the existing ones). If you decide to add symbols, all the necessary references will have to be modified.
3. You may wish to amend the payoff rates.
4. You may wish to change the musical accompaniment.

Appendixes

- Appendix A: Saving Your Programs on a Tape Recorder
- Appendix B: Keywords
- Appendix C: Graphics Symbols
- Appendix D: Keyboards

Appendix A

SAVING YOUR PROGRAMS ON A TAPE RECORDER

- Saving a Program
- Loading a Program
- Important Reminders

Typing in a long program each time it is needed can be very time-consuming. Moreover, having to key in a program each time leaves open the door to error. However, a program can be saved on magnetic tape in much the same way that you ordinarily save a piece of music by recording it on your tape recorder.

Saving a T/S 1000 or T/S 1500 Program

To save a program after it has been typed into your computer and checked out to be correct, all you have to do is to press the S key (**SAVE**) and enter a name for your particular program. If you choose the name "FLASH," for example, you would type into your computer:

SAVE "FLASH"

At this point you should start the recorder in its "recording" mode. Then hit the ENTER key, and the program (or more accurately, a copy of your program, since the program will still reside in the memory of the computer) will be recorded on the tape. You will know when the recording is complete by the termination code 0/0 which will appear at the

bottom left-hand corner of the screen. At that point you may turn off the recorder.

Saving a T/S 2068 Program

Once a program is typed into your computer (and checked for accuracy), it can be saved by pressing the S key (**SAVE**) and entering the name you have given your program. For example, enter the command:

SAVE "COLOR ORGAN"

Then hit the ENTER key. The T/S 2068 responds by displaying a message at the bottom of the screen. It instructs you to start your recorder, then hit any key. Once you do so, the border changes color until the save is complete. If everything went well it will display 0:OK.

If you wish to be sure that the program has been saved, then you must rewind the tape and type:

VERIFY "COLOR ORGAN"

Then play the recorder and hit the ENTER key. The T/S 2068 then compares what is on the tape to what is in its memory. A message 0:OK at completion indicates that the program was properly saved. If an error message occurs, repeat the **SAVE** procedure at a different volume.

Loading a T/S 1000 or T/S 1500 Program

In order to load a program from magnetic tape into the computer, type in **NEW** followed by the ENTER key. Now you type either:

LOAD "FLASH"

which searches for and loads the program called FLASH or:

LOAD " "

which searches for and loads the first complete program found on the tape.

Loading a T/S 2068 Program

To load a program into the T/S 2608 computer, enter the command:

LOAD "COLOR ORGAN"

Then play the recorder and hit the ENTER key. The border will begin to change colors. The T/S 2068 will list the names of any programs it encounters on the tape but will not actually load until a name matches the one specified in the **LOAD** command. After a successful load, the 0:OK message will be displayed. You can also use the command:

LOAD " "

which will load the first complete program it encounters on your tape.

Important Reminders

1. Don't forget to make a note of the *precise* name of the program you are storing, including any spaces between words that may exist in the name. If you were to save a program called "MASTERMIND" on tape and then later try to load a program called "MASTER MIND" from the same tape, the computer will *not* be able to find the program.
2. We have found that for either loading or saving a program, a tape recorder in the "mono" (rather than "stereo") mode works best. If your tape recorder is stereo only, you should turn the "balance" control all the way left or all the way right. In addition, you should leave the "volume" control quite high, perhaps three-quarters of the way to full volume; the "tone" control should likewise be at its highest (treble) setting.

Appendix B

KEYWORDS

Keywords for the T/S 1000 and T/S 1500

The following words, listed in alphabetical order, can be accessed on both the T/S 1000 and the T/S 1500 with a single keystroke. These keywords always appear in **BOLDFACE** in the program listings in this book.

Keyword	Mode	Key	Meaning
ABS	F	G	absolute value
ACS	F	S	arcosine
AND	K or L	shifted-2	boolean connector
ASN	F	A	arcsine
AT	F	C	specify print position
ATN	F	D	arctangent
CHR$	F	U	number to character
CLEAR	K	X	clears variables and screen
CLS	K	V	clears screen
CODE	F	I	character to number
CONT	K	C	continue execution
COPY	K	Z	copies screen to printer
COS	F	W	cosine
DIM	K	D	dimensions arrays and strings
EXP	F	X	exponential

Keyword	Mode	Key	Meaning
FAST	K or L	shifted-F	fast mode
FOR	K	F	sets up loop
GOSUB	K	H	subroutine branch
GOTO	K	G	unconditional branch
IF	K	U	condition testing
INKEY$	F	B	keyboard read
INPUT	K	I	accepts data
INT	F	R	returns integer portion
LEN	F	K	length of string
LET	K	L	assigning values
LIST	K	K	lists program
LLIST	K or L	shifted-G	lists on printer
LN	F	Z	natural log
LOAD	K	J	loads program from tape
LPRINT	K or L	shifted-S	prints to printer
NEW	K	A	clears memory
NEXT	K	N	indicates end of loop
NOT	F	N	negation
OR	K or L	shifted-W	boolean connector
PAUSE	K	M	pauses the machine
PEEK	F	O	shows memory at specified point
PI	F	M	value of π
PLOT	K	Q	places pixel on screen
POKE	K	O	places data in specified point of memory
PRINT	K	P	prints item
RAND	K	T	indicates starting position of **RND**
REM	K	E	remarks
RETURN	K	Y	return to **GOSUB** call
RND	F	T	random number generator

Keyword	Mode	Key	Meaning
RUN	K	R	runs program
SAVE	K	S	saves to tape
SCROLL	K	B	scrolls one line
SGN	F	F	sign of a number
SIN	F	Q	sine
SLOW	K or L	shifted-D	slow mode
SQR	F	H	square root
STEP	K or L	shifted-E	increment of **FOR/NEXT**
STOP	K or L	shifted-A	stops execution
STR$	F	Y	numeric values to string
TAB	F	P	tabs to a column
TAN	F	E	tangent
THEN	K or L	shifted-3	resultant of a true condition
TO	K or L	shifted-4	specifies boundaries
UNPLOT	K	W	unplots pixel
USR	F	L	machine code within BASIC
VAL	F	J	converts numeric characters to actual numeric values

Keywords for the T/S 2068

The list below shows the keywords for the T/S 2068 model. Note that there are several new keywords and that the shifted keywords are accessed by a (C)shift (caps shift—white key) or an (S)shift (symbol shift—black key).

Keyword	Mode	Key	Meaning
ABS	E	G	absolute value
ACS	E	(S)shifted-W	arcosine
AND	K, L, or C	(S)shifted-Y	boolean connector

Keyword	Mode	Key	Meaning
ASN	E	(S)shifted-Q	arcsine
AT	K, L, or C	(S)shifted-I	specify print position
ATN	E	(S)shifted-E	arctangent
ATTR	E	(S)shifted-L	attributes
BEEP	E	(S)shifted-Z	sound
BIN	E	B	binary
BORDR	K	B	border
BREAK		BREAK key	breaks a running process
BRIGHT	E	(S)shifted-B	changes intensity of display
CAPS LOCK	L or C	(C)shifted-2	changes letter mode
CAT	E	(S)shifted-9	catalog
CHR$	E	U	number to character
CIRCLE	E	(S)shifted-H	draws a circle
CLEAR	K	X	clears
CLOSE#	E	(S)shifted-5	closes file
CLS	K	V	clears screen
CODE	E	I	character to number
CONT	K	C	continue execution
COPY	K	Z	copies screen to printer
COS	E	W	cosine
DATA	E	D	data statement
DEF FN	E	(S)shifted-1	define function
DELETE	K, L, or C	(C)shifted-0	deletes character or line
DIM	K	D	dimensions arrays and strings
DRAW	K	W	draws a line
EDIT	K	(C)shifted-1	edits a line
ERASE	E	(S)shifted-7	erases a file
EXP	E	K	exponential
FLASH	E	(S)shifted-V	flash attribute
FN	E	(S)shifted-2	function
FOR	K	F	sets up loop
FORMAT	E	(S)shifted-O	formats a file

Keyword	Mode	Key	Meaning
FREE	E	(S)shifted-A	number of bytes free
GO SUB	K	H	subroutine branch (note: two words)
GO TO	K	G	unconditional branch (note: two words)
GRAPHICS	K, L, or C	(C)shifted-9	get into graphics mode
IF	K	U	conditional testing
IN	E	(S)shifted-I	in
INK	E	(S)shifted-X	ink (letter) color
INKEY$	E	N	keyboard read
INPUT	K	I	accepts data
INT	E	R	returns integer portion
INVERSE VIDEO	K, L, or C	(C)shifted-4	inverse video
INVERSE	E	(S)shifted-M	inverts paper and ink
LEN	E	K	length of a string
LET	K	L	assigning values
LINE	E	(S)shifted-3	line command
LIST	K	K	lists program
LLIST	E	V	lists on printer
LN	E	Z	natural log
LOAD	K	J	loads program from tape
LPRINT	E	C	prints to printer
MERGE	E	(S)shifted-T	merge files
MOVE	E	(S)shifted-6	move
NEW	K	A	clears memory
NEXT	K	N	indicates end of loop
NOT	K, L, or C	(S)shifted-S	negation
ON ERR	E	(S)shifted-F	error flag set
OPEN#	E	(S)shifted-4	opens a file

Keyword	Mode	Key	Meaning
OR	K, L, or C	(S)shifted-U	boolean connector
OUT	E	(S)shifted-O	out designation
OVER	E	(S)shifted-N	overstrike
PAPER	E	(S)shifted-C	background color
PAUSE	K	M	pauses the machine
PEEK	E	O	shows memory at specified point
PI	E	M	value of π
PLOT	K	Q	plots a point
POINT	E	(S)shifted-8	coordinates of a point
POKE	K	O	places data into specified point of memory
PRINT	K	P	prints item
RAND	K	T	indicates starting position of **RND**
READ	E	A	reads from data statement
REM	K	E	remarks
RESET	E	(S)shifted-P	resets attributes
RESTORE	E	S	restores data statements
RETRN	K	Y	return from a subroutine
RND	E	T	random number generator
RUN	K	R	runs program
SAVE	K	S	saves to tape
SCREEN$	E	(S)shifted-K	used in screen manipulation
SGN	E	F	sign of a number
SIN	E	Q	sine
SOUND	E	(S)shifted-G	invokes sound synthesizer
SQR	E	H	square root

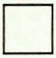
Keyword	**Mode**	**Key**	**Meaning**
STEP	K, L, or C	(S)shifted-D	increment of **FOR/ NEXT**
STICK	E	(S)shifted-S	tests joystick
STOP	K, L, or C	(S)shifted-A	stops execution
STR$	E	Y	numeric values to string
TAB	E	P	tabs to a column
TAN	E	E	tangent
THEN	K, L, or C	G	resultant of a true condition
TO	K, L, or C	(S)shifted-F	specifies boundaries
TRUE VIDEO	K, L, or C	(C)shifted-3	true video
USR	E	L	machine code within BASIC
VAL	E	J	converts numeric characters to actual numeric values
VERIFY	E	(S)shifted-R	verifies a save operation

Appendix C

GRAPHICS SYMBOLS

T/S 1000 and T/S 1500 Graphics Symbols

Please note that in this listing each symbol is "boxed," meaning that all four of its borders are shown even though these borders will *not* appear on the screen. For example, the very first symbol listed here will show up on the screen merely as a space, and not an empty box. It should also be noted that letters in inverse video are also accessed via the G mode.

Symbol	Mode	Key	Symbol	Mode	Key
	K or L	SPACE		G	SPACE
	G	shifted-1		G	shifted-Q
	G	shifted-2		G	shifted-W
	G	shifted-7		G	shifted-6
	G	shifted-4		G	shifted-R
	G	shifted-5		G	shifted-8
	G	shifted-T		G	shifted-Y
	G	shifted-E		G	shifted-3
	G	shifted-A		G	shifted-H
	G	shifted-D		G	shifted-G
	G	shifted-S		G	shifted-F

T/S 2068 Graphics Symbols

Use the following list whenever you are working with the T/S 2068; it differs from the list given for the T/S 1000 and T/S 1500. Note that six of the symbols available on the T/S 1000 and T/S 1500 are not available on the T/S 2068. If you are using a game from Part I or Part II of this book on the T/S 2068 and one of these six symbols appears, just substitute a symbol from the list below.

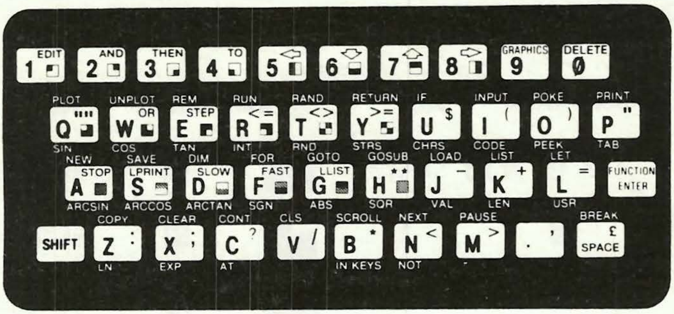
Note that when accessing the graphics symbols on the T/S 2068, some of the keys must be hit while holding down the shift key, while others are accessed in the nonshift position. Either the caps shift or the symbol shift can be used. Letters in inverse video are *not* accessed via the G mode on the T/S 2068; the inverse key is used for that purpose.

Symbol	Mode	Key	Symbol	Mode	Key
(blank)	K, L, or G	SPACE	█	G	shifted-8
▘	G	2	▟	G	shifted-2
▝	G	1	▙	G	shifted-1
▀	G	3	▄	G	shifted-3
▖	G	shifted-7	▜	G	7
▌	G	shifted-5	▐	G	5
▞	G	shifted-6	▚	G	6
▛	G	shifted-4	▗	G	4

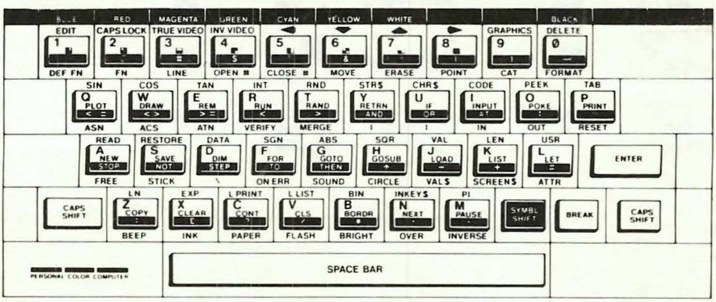
Appendix D

KEYBOARDS

The T/S 1000 and T/S 1500 Keyboard



The T/S 2068 Keyboard



Computers

MUNCHERS

The Ultimate **Timex/Sinclair** Games Book

Yin Chiu and Henry Mullish

Here are 32 entertaining new games programs written in Timex/Sinclair BASIC. These fun and exciting games are written for all the Timex/Sinclair home computers, including the classic 1000, the new 1500 Personal Computer, and the 2068 Color Computer. All the programs have been fully tested and will provide countless hours of entertainment for the entire family. For the absolute beginner, there are explicit, step-by-step instructions for keyboarding the programs. And for the more experienced reader, there are suggestions on how to modify and expand the programs. MUNCHERS is the ideal introduction to the Timex/Sinclair 1000, 1500, and 2068 computers—three of the most popular personal computers ever made.

Yin Chiu is the coauthor of *CRUNCHERS: 21 Simple Games for the Timex/Sinclair 1000.* He is a systems analyst for a major Wall Street firm.

Henry Mullish teaches computer science at New York University's Courant Institute of Computer and Mathematical Sciences. He is the author of more than fifteen books on computers and programming, including *CRUNCHERS.*

McGraw-Hill

ISBN 0-07-010839-0